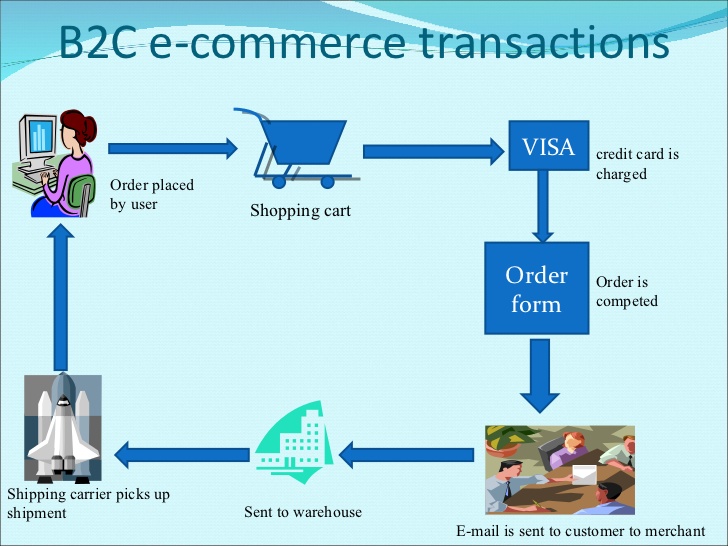


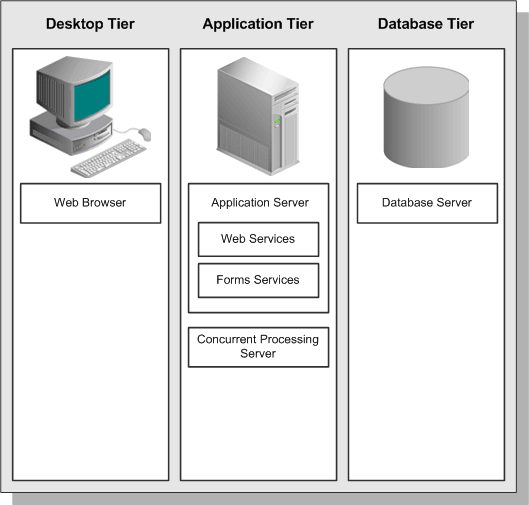
**Problem Description**

“EzeeShop” is an online shopping portal which helps millions of shoppers browse, discover and purchase items in an easy, fast and intuitive manner. EzeeShop follows the B2C Model of bringing Business to Customers. Company has been profitable for many years (till last year) and was a monopoly in the market because of its exclusive agreement with many leading couriers.

For five years in a row, “EzeeShop” has been the most popular Website in Singapore, with more than 82% market share. But, for its investments and revenues, the current visibility in the market is not enough. It must attract more advertisers and create services for which users are willing to pay. Company needs to understand what its visitors want to buy and what information they want to see.

“EzeeShop” Sales Director, decided to increase the company’s revenues. He needed to analyse entire data and create customer profiles that would optimize the selection and placement of advertising banners. “EzeeShop” has large volumes of data collected over last 10 years and since it is one of the most popular websites, it has long lasting customers.

**Understanding Existing System Architecture**

1. “EzeeShop” currently has a traditional web architecture consisting of 3 tiers.
2. First tier is the Desktop / Client tier where the client places his request through HTML for HTML-based applications or via a Java applet in a Web browser for the traditional Forms-based applications.
3. Second tier is the Application tier, which has a dual role: hosting the various servers and service groups that process the business logic, and managing communication between the desktop tier and the database tier.
4. Third tier is the Database tier, which could be any data storage system such as RDBMS. The database does not communicate directly with the desktop clients, but rather with the servers on the application tier, which mediate the communications between the database server and the clients.

**Scope of Project**

Scope of the project is to increase revenue of “EzeeShop” by implementing a solution which will analyze the huge volume of customer data and create customer profiles that would optimize the selection and placement of advertising banners. Solution provided is not to replace the current OLTP system which is still required to be in place to provide service to their customers. The system proposed is to compliment the current system with Big data technologies with Hadoop Ecosystem.

The existing system has limitations that is unable to meet “EzeeShop” new business needs of analyzing huge datasets, creating customer profiles and optimize the selection of advertising banners. With the implementation of Big Data Technologies, these limitations are overcome as it is capable of storing and processing huge volumes of different types of data. This not only enables “EzeeShop” to fully utilize its huge dataset, but is now able to integrate and analyze external sources like social media (tweets, facebook posts, chats etc), internal unstructured data (eg emails) and web data (click stream, web logs, website activity etc). Thus with this multiple sources of data, “EzeeShop” can better understand its customers. Keeping in mind the business objectives, this facilitates creation of customer profiles, and selection of appropriate advertisement banners to be displayed to the customers.

This project aims to introduce the Hadoop ecosystem and its components (eg Hive, HBase), to offer the advantages of Big Data Technologies without compromising the advantages/features of DW technology. For example, Hadoop cannot handle transactions (random access to rows of data), cannot handle non-parallelized work, cannot process lots of small files and cannot perform intensive calculations with little data or small files. These task can be performed by the DW. Hadoop is designed for large files (datasets), provides scaling (linearity of performance even as the volume of data grows), analysis and summary of huge volumes of data, and handle unstructured/structured data from various sources. These tasks cannot be performed by the DW.

**Understanding of Current Architecture**

“EzeeShop” is embarking on a typical traditional Data Warehouse Architecture to support it’s operations and business activities via the web interface. The current DW handles mostly structured data from a few internally generated data sources such as Customer (database of customer info), Provisioning (warehousing/logistics) and Billing (transaction of purchases) in the form of OLTP, CRM systems. The data is then accumulated in a Data warehouse staging area where the raw data is stored before it is imported into the DW itself. The data is then imported, refined and processed with ETL tools and stored in the DW which it typically on Oracle. Within the DW there are Datamarts (ex the HR department will have it’s own Datamart in the DW). This processed and aggregated data is then exported out of the DW, via an ETL tool. The OLAP allows slicing/dicing of data and enables drill down/drill up capabilities to enable easy visualization of the data and allows enterprise decision making. In the next stage, the data is visualized via BI tools, analyzed by statistical software for trends (descriptive analytics), mined by Datamining algorithms for insights/patterns and some of this data may be displayed on the “EzzeShop” website or to investors/stakeholders.

There are limitations to the DW architecture. Firstly, the data being used are all internally generated and are in structured format (typically spreadsheets or text files), the information is usually imported into the DW by batch processing. There is no processing of current (real-time data), nor processing of other data sources (eg unstructured data, web data, social media data). Secondly, as the size of the database increases (more data being generated over time), the performance of the DW decreases, due to it’s non-parallel distribution of data and the movement of data to the processing engine can increase network congestion and create ‘bottlenecks’ especially if we are dealing with large volumes of data. Thirdly, only historical data is available for data analysis and datamining, thus the DW is limited to descriptive analytics and making predictive analytics difficult. Thus decision making is limited to internal data sources, with limited performance and limited predictive capabilities.

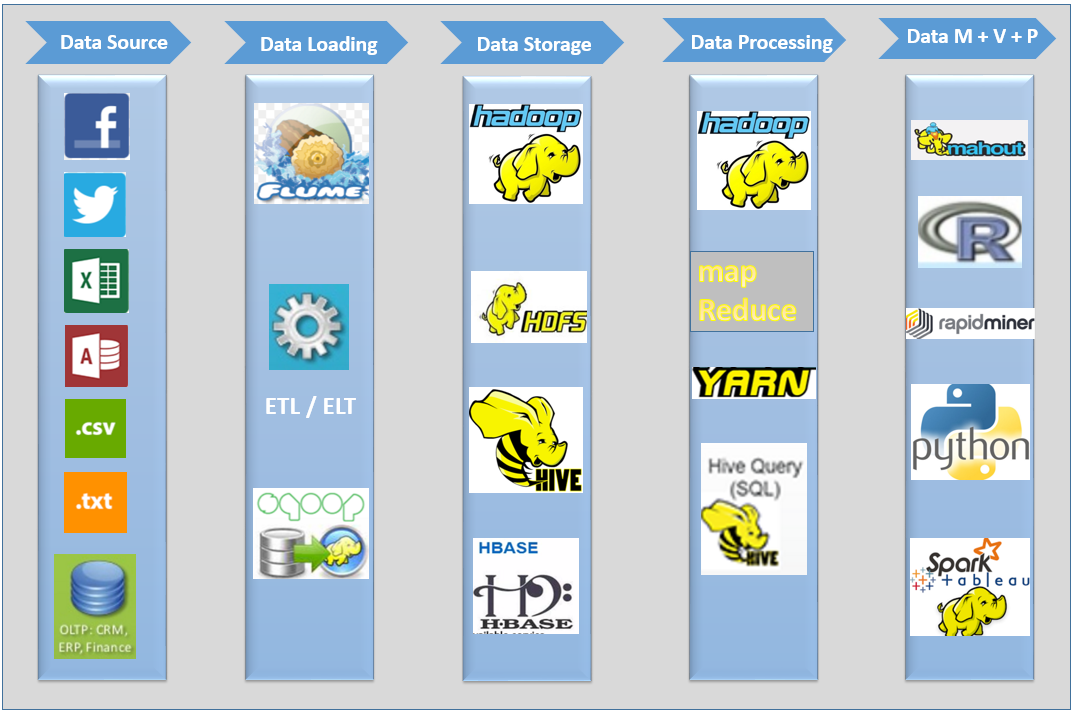
**Proposed Architecture (Pain Points elimination)**

The proposed system is to use Hadoop which enables in an increase in capabilities of handling large Volumes of data, high Velocity of data (batch and streaming data), different Variety of data (structured spreadsheets & text files and unstructured documents such as emails, audio, graphics, videos), Veracity of data (various quality and certainty of data – process and integrity checking) and finally Value of data (generate insights from the data). BDT uses data clusters and parallel distribution and processing of data to solve the problem of lagging performance. The data is broken into (64/128 MB blocks) and is replicated and distributed across the nodes. This allows high availability, high scalability and high flexibility of data. Moreover, the processing involves moving the logic to where the data resides (distributed) and retrieving the aggregated results (via Map Reduce), this maximize the utilization of the network bandwidth and prevents ‘bottlenecks’.

To solve the problem of limitations of data structure, BDT facilitates the analysis of unstructured data from external source that provide EzeeShop useful info into their customers. Social media and other online web sources are a rich form of information that can provide business value. By tapping into this data, BDT allows more meaningful insights into their customers purchasing habits, sentiment analysis beyond their internal records. Thus enterprise decision making is more informed and based on more data.

To solve the problem of lack of streaming data and predictive analytics, BDT allows real-time flow of data via web API and other sources to create ‘current data’, while machine learning algorithms (ML) such as clustering and classification can be performed on “EzeeShop” existing dataset and act as the ‘training data’. By performing the ML algorithms on these two sets of data (current and training). “EzeeShop” can generate useful insight and categorization of data/profiling of customers to meet it’s business objectives. As more data streams in, the ML algorithms can be updated to provide more accurate analysis of ‘current’ data more accurately select appropriate advertisement banners.

**Proposed Model**



The proposed model, consists of many layers of operations, the first layer is the Data Source Layer which comprises of different types of data (structured/unstructured), different forms of data (streaming/batch), different modes of data (web, external, internal) and inclusive of the traditional internal data sources of the DW. By having a rich variety of data sources, “EzeeShop” can have more information about their customers, the market and their business operations and opportunities.

The second layer is the Data Loading Layer which imports these data sources (different types and structures) into the system using tools such as Sqoop and Flume. Typically the data varies by images, videos, audio, emails (text), website logs, API stream and so on, which is imported into the system.

The third and fourth layer is the Data Storage and Processing layer which uses HDFS and MapReduce of Hadoop for Storage and Processing. The output is typically stored in read only database, with the updates completely replacing existing precomputed views. The data is typically stored in a fast NoSQL database. The serving layer is where the output of the batch and speed layer is stored and responds to ad-hoc queries by returning precomputed views or building views from the processed data.

The fifth layer is the combination of Data Mining, Data Visualization and Data Presentation where ML algorithms are carried out by Mahout (primarily for classification and clustering) to create customer profiles, based on the Big Data dataset. Rapid Miner (Radoop) is used for data mining and to perform predictive analytics on the Big Data dataset. Data Presentation comprises of tools for the three areas of Data Analytics, BI reporting and Data Visualization. Here data is analyzed & mined for patterns, trends & useful insights and later presented to stakeholders for enterprise decision making. Data for customers is displayed though the website interface. Keeping in mind the business objectives, the large data from the previous layer is processed, categorized and summarized (as described above) to optimize the selection of advertisement banners which is then displayed to selected customer profiles. This layer is the output of the BDT.

**Considerations for Hadoop**

Hadoop primarily handles batch processing, consisting of HDFS (data storage), MapReduce (data processing) and YARN (framework for job scheduling and cluster resource management). There is a growing need to store increasing volumes of data which keeps growing every year. Hadoop allows addition of clusters/nodes to incorporate the growing data needs of “EzeeShop” due to it’s scalability. Data is replicated, backed up and distributed across the nodes, thus it is made highly available. If there is any shortage of storage space, more data nodes can be setup to increase storage capacity and linked to the cluster without compromising it’s operations. There is also a need to increase processing power, in order to process this large volumes of data, processing is done in parallel and the logic is distributed across the clusters (via mapreduce). The master metaserver (name node) manages and monitors the clusters, and update the info in the clusters (slaves – data nodes) as well as stores the metadata. The masterserver (jobtracker) distributes the Mapreduce jobs to all the slave servers (task trackers) which individually processes the data stored in the data node using this logic, and sends the aggregated results back to the jobtracker. This method of moving the logic to the stored data location and mapping back the results, makes BDT ten times faster than DW which moves the data to the logic (processing engine).

There is a need for availability of data, the large data files are broken down into blocks of 64MB/128MB are replicated (minimum of 3 times) and distributed across the data nodes in the cluster, thus we have 3 copies of each data block in different locations. In the event of any fault to one of the nodes, Hadoop is fault tolerant and self-healing, the data is not lost as it is replicated across the other nodes. The data node constantly sends a ‘block report’ to the name node to inform the name node of it’s availability. A working loss/breakdown of any of the slave nodes does not compromise the operation of the entire cluster.

There is a need to query the data, Hadoop is optimized for streaming access to large files and follow the write once, read many ideology. There is a need to be cost-effective, Hadoop allows utilization of commodity hardware and is open-source software.

**Considerations for Hive**

HIVE which runs on top of HDFS/Hadoop is a data warehouse software primarily for querying data and appending data. HIVE is a SQL like engine which runs MapReduce jobs. HIVE provides a SQL-like interface database query (called HiveQL), it is user-friendly and familiar to SQL-users. HiveQL converts queries into MapReduce and Spark jobs which is submitted to Hadoop for execution. HIVE is typically used for long batch-processing jobs such as log parsing, ETL, etc, and not for real time operations.

There is a need to append data generated by business and operational activities, HIVE facilitates the appending of large amounts of data (bulk loading) to a dataset. The SQL interface is a schema on read (doesn’t check the data type) and is dynamic flexible schema (no predefined schema) thus allows faster and more efficient loading. HIVE does not support row-by-row update (only support bulk loading) and therefore suitable for batch jobs.

HIVE acts as a warehouse directory and supports the analysis of large datasets stored in HDFS, as well as provides a tool to enable easy data ETL. HIVE is much faster than SQL and has a high level of abstraction, it has less line of code than PIG or MapReduce. HIVE provides data summarization, analysis and ad-hoc querying and is more suited for data collected over a period of time. In the case of Ezeeshop, it has an extensive customer dataset which is suited for querying with HIVE. HIVE supports only four file types (textfile, sequence file, orc file, rc file), but can also be mapped to existing HBase tables.

HiveSQL allows interface like SQL which is familiar to many programmers and DBAs and HiveSQL also support cross record transaction as well as complicated access patterns (such as joins), HIVE supports full table scans and doesn’t use index access (eg HBase). HIVE can be learnt easily; however, HIVE can only process structured data (operates like SQL).

**Considerations for HBase**

HBase is a distributed, scalable NoSQL key/value database (runs on top of HDFS/Hadoop) with real-time access, to very large tables (supports millions of rows and columns) of sparse data. HBase provide Bigtable like capabilities on top to Hadoop and HDFS. HBase is a schema on read (doesn’t check the data type) and is dynamic flexible schema (no predefined schema), and it supports strictly consistent and fast read/writes operations as well as realtime random access to the data for querying. HBase allows automatic and configurable sharding of tables across the servers (also called ‘shards’).

There is a need of flexibility of data, “EzeeShop” may obtain new data from various data sources and new categories of data may be introduced as well. Thus new columns and rows must be added to existing dataset table. HBase schema consists of a pre-defined table schema with column families. Column families allow many attributes to be grouped together, and the elements of the column family are stored together. However, HBase is flexible as new columns can be added to families at any time, making the schema flexible and able to adapt to changing application requirements.

There is a need to query and access sparse data, this is common in most bigdata cases. Sparse data is defined as small amounts of information caught within collection of empty/unimportant data – typically 0.1% of huge data collection). HBase allow fast definition, fast low-latency access to data which can be done in real time and randomly (lookup of individual records), in a huge dataset.

There is a need to query real-time data. HBase fits for real-time querying of Bigdata and real-time analytics and is ideal for fast transactional systems (unlike HIVE which codes for MapReduce jobs – batch processes), due to it’s consistent and fast random read/write access to the data and storage of data in key/value pairs. HBase allows indexing and enables faster definition of data, thus enabling very fast access to data (if you know the key you can instantly get the value).

**Components of the Model**

The main components of Hadoop ecosystem is HDFS and MapReduce which are used for data storage and data processing respectively; HIVE and HBase are two different Hadoop technologies (part of Hadoop ecosystem) which can be used together, where data can be read and written from HIVE to HBase and back again. Sqoop and Flume deal with importing structured and unstructured data respectively into the Data Operations layer. Sqoop is an interface to import data from relational databases into Hadoop to populate tables in Hive or HBase. Flume is primarily used to collecting and moving log data.

Mahout implements ML algorithms focused on clustering and classification to provide predictive analytics capabilities on Bigdata sets and ‘current’ data.

Other Hadoop ecosystems involved are Zookeeper and Storm etc. Zookeeper is configuration & synchronization service and acts as a naming registry for Hadoop and stores metatdata (table locations in HBase). The YARN service registry is built on top of Zookeeper and allows multiple nodes to perform together. Oozie is a workflow scheduling system (control flows) to manage Hadoop jobs and trigger the execution of computation tasks.

Storm allows batch, distributed processing of streaming data while Spark can run programs up to 100X faster than Hadoop Mapreduce in memory (10X faster on disk) and is used for analytics on streaming data. Both Spark and Storm can be used with a variety of programming languages and can be used to access HBase and have a variety of applications.

**Feasibility of the Model**

HDFS and Map Reduce are the essentials of the BDT architecture to allow storage, distribution and processing of data among the clusters. As “EzeeShop” data requirements grow, these two Hadoop technologies are necessary to provide flexible & scalable distribution of data and provide parallel processing to avoid a lag of performance.

Sqoop, and Flume provide “EzeeShop” with the capability to import multiple data types (unstructured/structured) from internal (eg. CRM, OLTP), external(eg. emails) or online sources (eg. social media). Thus there is more data for Ezeeshop to perform analytics and data mining algorithms to generate useful insights about their customers for decision making.

Zookeeper is necessary for scheduling and synchronization of jobs, it allows multiple nodes to be coordinated together correctly and is necessary for complex data operations of Hadoop.

Mahout ML capabilities is necessary for the selection of banners (Ezeeshop business objectives) based on customer profiling, upon analytics of datasets and current data.

Storm and Spark fast computational power allows processing of streaming data, to obtain the most up-to-date information about customers and to ‘update’ ML algorithms (for selection of advertisement banners) and perform data analytics on the data to facilitate “EzeeShop” decision making.

**Assumptions made in the Model**

This model assumes that “EzeeShop” wants to gain the competitive advantage in the marketplace by taking advantage of the complex Hadoop ecosystem and utilize as many of it’s open source software as possible. This model incorporates such various software and allows “EzeeShop” to gain access to unstructured data, messaging system, multiple sources of data (web/external), including streaming data. It is assumed that “EzeeShop” will eventually process (aggregate and perform ML algorithms) and use this complex dataset for data analytics, business intelligence (BI) and selection of advertisement banners.

This model also assumes that “EzeeShop” already has an extensive data collection and that it’s data requirements will grow at a fast rate due to business operations, therefore the need for scalability (addition of more data nodes) and parallel processing (via MapReduce). Zookeeper is necessary for coordination of all these services and management of data as well as storage of metadata.

This model also assumes that “EzeeShop” has a need for BI and data analytics, which is needed to maintain it’s competitive edge. Querying tools such as HIVE and elastic search are included to provide such capabilities and to work in conjunction to BI tools such as Tableau. Aggregation and summarization of data can provide vital statistics to stakeholder and can be displayed on “EzeeShop” website for customers to see (for better customer service or information about the company – if applicable). Businesses are looking for ways to mine real-time insights to provide competitive advantages, the BDT model provides “EzeeShop” such an advantage.

**Merits of the Proposed Model**

As discussed, the Big Data Technology is meant to complement and not replace the existing “EzeeShop” IT System, it facilitates the growing needs of storage capacity and processing power for “EzeeShop” growing database. Both the Big Data Technology and the “EzeeShop” existing DW system share the same BI intelligence system layer, thus information can be combined, integrated and processed together in this layer.

Predictive analytics and Machine Learning capabilities are possible in the proposed model, streaming data in real-time allows for the visualization and analysis of current data as well as updating of the algorithms to allow forecasting, optimizing decision making as well as transaction profiling. “EzeeShop” is able to foresee & capitalize on trends, avert potential problems and make decisions and take actions early (including risk management and customer satisfaction).

The proposed system, allows multiple business operations, data mining capabilities on a huge dataset, thus allowing discovery of hidden patterns and utilization of ‘dark data’. Data scientists and data analysts have a huge dataset at their disposal and a variety of datamining tools to perform ad-hoc analysis and are not limited to propose projects from management. This can reveal new discoveries about business operations, customer behaviour and so on which will facilitate decision making.

**Project Implementation and Application Deployment**

“EzeeShop” now has the capabilities to store, process and analyze it’s huge dataset using BDT to allow selection of advertisement banners that will attract customers and promote sales. Once the project is implemented, careful monitoring of the customers who respond to these banners (clicking on them) and actual purchases of products should be recorded and monitored. Data mining algorithms should be constantly checked for it’s efficiency and performance, although much has been considered to provide this service to customers, the ML algorithms needs to be updated and reviewed constantly and changed if necessary.

After sales service/surveys on customer satisfaction and feedback about the advertisement provide a source of rich information that can be useful to Ezeeshop to fine tune it’s algorithms and selection of banners. Also as the ‘training’ data grows and the ‘current’ data changes, these can be used to further update the algorithms.

**Short-comings of the Proposed Solution**

The model uses HBase as the database to access HDFS files (batch processing). HBase depends on Zookeeper, Namenode and HDFS for it’s operations and performance and is complex to maintain. Indexes are available only on row key and it is weak for sequential data scans and has no ACID compliance. The master/slave architecture of Hadoop is mimicked in HBase and has it’s drawbacks if the master fails.

The threat imposed on a single point of failure on the Namenode is high, even with a secondary Namenode at hand, operations can be delayed if such an event occurs. The model is highly complex given the complexity of Hadoop clusters, making data management and security, issues to consider. Hadoop was developed with poor security considerations and has a weak and complex security model without data encryption and has limited authorization capabilities. The distributed nature of Hadoop makes security difficult.

**Recommendations to address the Short-comings**

To address the issue of limitation to data store options, Cassandra (a popular NoSQL software) can be considered as an alternative or compliment to HBase (since it can also be integrated with Hadoop and Spark). Cassandra is faster than HBase and is suited for time series data (streaming data). Cassandra performance scales linearly as you add servers and is easier to manage than HBase and offers lots of control of how data is stored on disk. Storage nodes are spread across multiple data centers and connected through high speed data link, all nodes are the same, there is no master/slave architecture as in HBase thus limiting the threat of breakdown. Overall Cassandra offers less operational overhead for better scalability than HBase. It is independent from Zookeeper and has it’s own file system, making it independent of HDFS. Cassandra is optimized for writes (more suited for streaming/machine data, time-series and transactional data) and single row queries while HBase is optimized for reads (more suited for range-based scans). Ezeeshop has a need to use the current data (streaming, real-time data) for it’s predictive analytics purposes which is directly tied to their business objectives and have more need for writes than reads (which Cassandra is suited for).