CHAPTER 6

Module VI

Data Analytics for IoE

Syllabus :

Introduction, Apache Hadoop, Using Hadoop MapReduce for Batch Data Analysis, Apache Oozie, Apache Spark, Apache Storm, Using Apache Storm for Real-time Data Analysis, Structural Health Monitoring Case Study, Tools for IoT:-Chef, Chef Case Studies, Puppet, Puppet Case Study -Multi-tier Deployment, NETCONF-YANG Case Studies, IoT Code Generator.

6.1 Introduction

Term “Big Data” gained momentum in early 2000s. Large amount of structured and unstructured data which cannot be processed by traditional relational database engines is referred as Big Data. Big data is complex such that traditional database engines are inadequate to process it. Open source framework Like Hadoop is the reason for growth of Big Data which was created to store and maintain large data sets. Big data involves generating, storing, analysing, retrieving and transferring large amount of data. The term “Big Data” not only refer to how much data you have but also what you do with that data. It involves various techniques, frameworks as well as tools. Big data helps to analyse data, identify patterns and behaviour which helps to take decisions in better manner which results in development of business or organization.

6.2 What is Big Data ?

Big Data is a large amount of data in which data analyze, visualize, summarize and extract the information using application software. It is characterized by Volume, Variety, Veracity, Velocity and Value.

6.2.1 Modern Corporate Need for BD Strategy

1. **Banking and Securities :** For monitoring financial activities and to reduce the fraudulent activities big data strategies are used. It is also used to analyze, process and monitor stock market data to predict the life of shares.
2. **Communication and Medias :** For real-time data, media industry we can use big data strategies to identify the latest trends in the market.
3. **Healthcare :** To gather public data and analyze, process, summarize the data for the benefits of society to identify solutions for diseases.
4. **Insurance :** For everything which handling new products can be claims through Predictive

analytics.

5. **Manufacturing :** To Increase productivity we can use big data by Supply chain management.
6. **Customer Trade :** To predict the inventory requirements on the basis of customer requirements. Many companies are using it by providing a loyalty card to keep a track of consumers.

6.2.2 Challenges in Big Data

Following are some challenges in handling big data :

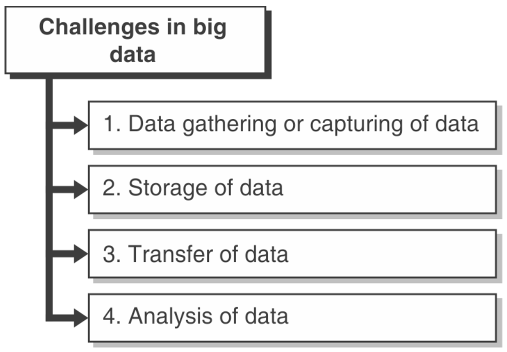


Fig. 6.2.1 : Challenges in big data

1. **Data gathering or capturing of data** : From different sources data is captured. Format of data is different. Data can be structured or unstructured or semi structured.
2. **Storage of data :** First problem in big data is obviously storage. Big data size is increasing very fast and as compared to size, storage capacity should also increase.
3. **Transfer of data :** One of the important and major in Big data is It's transfer as different systems are associated in data transfer.
4. **Analysis of data :** Processing of data and its analysis required too many calculations. Processing of data depends on software used like Operating system, memory size and processing power of the system.

6.3 Types of Data

1. Structured

By structured data, we mean data that can be processed, stored, and retrieved in a fixed format. It refers to highly organized information that can be readily and seamlessly stored and accessed from a database by simple search engine algorithms. **For instance, the employee table in a company database will be structured as the employee details, their job positions, their salaries, etc.,** will be present in an organized manner.

2. Unstructured

Unstructured data refers to the data that lacks any specific form or structure whatsoever. This makes it very difficult and time-consuming to process and analyze unstructured data. Email is an example of unstructured data.

3. Semi-structured

Semi-structured data pertains to the data containing both the formats mentioned above, that is, structured and unstructured data. To be precise, it refers to the data that although has not been classified under a particular repository (database), yet contains vital information or tags that segregate individual elements within the data.

6.4 Characteristics of Big Data

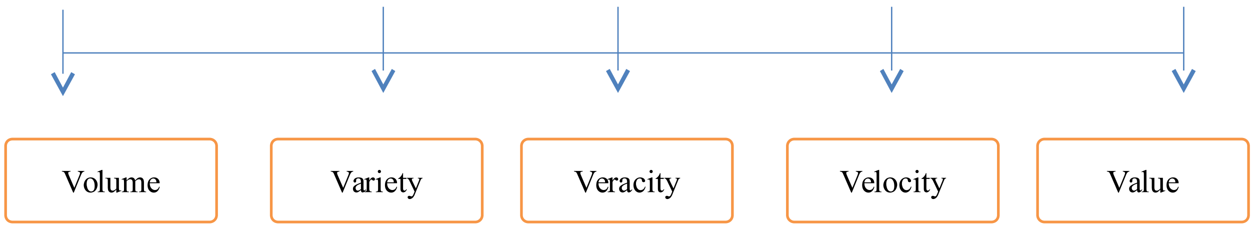


Fig. 6.4.1 : Characteristics of big data

- Back in 2001, Gartner analyst Doug Laney listed the **3 'V's of Big Data – Variety, Velocity, and Volume.**
- These characteristics, isolated, are enough to know what is big data. Let's look at them in depth:
- **Volume :** It is the quantity of generated or stored data. Data can be produced by various ways like the stock market, airline systems, Natural systems, Transportation, manufacturing, etc. Data size is transformed from zettabyte to petabyte.
- **Variety :** Data is produced in various formats. It may be structured, Unstructured and semi structured data. With traditional systems, it is not possible to manage a large amount of data. Structured data is a data-oriented in one particular format like tabular formats having rows and columns, Unstructured data is a combination of text, audio, images, videos. And semi-structured data is a data which is separated by a particular separator like a comma, tab.
- **Velocity :** In today's era data is transferred from batch processing data into streaming data. Speed of data generation or the processing is similar to "velocity of light" it increases on a continuous basis. Compared to small data, Big data is produced continuously.
- **Veracity :** Veracity is nothing but an Uncertainty of data. It refers to Data Quality and Data Value.
- **Value :** Generated data from the diverse sources is raw data. Value of the data can be discovered by analyzing data, recognizing patterns and by predicting behavior.

6.5 Big Data Handling Techniques

Handling of Big Data is another major concern. Below are some emerging technologies that are helping users cope with and handle Big Data in a cost-effective manner. Big data handling can be done with respect to following aspects :

- **Processing Big data :** MapReduce, Hadoop is an integrated framework for processing and storing Big data
- **Analysis and querying of data :** WibiData, PLATFORA, PIG
- **Business Intelligence :** Hive
- **Storage :** Cloud storage, Column-oriented databases, schema-less databases

- **Machine Learning :** Apache Mahout, SkyTree Some of the various Big data handling techniques defined are illustrated below

1. MapReduce for batch data analysis

- MapReduce is the key algorithm that the Hadoop MapReduce engine uses to distribute work around a cluster. Mapper function-A map transform function is provided to transform an input data row of key and value to an output key/value:
  - map(key1,value) -> list<key2,value2>That is, for an input it returns a list containing zero or more (key, value) pairs: The output can be a different key from the input. The output can have multiple entries with the same keyReduce function : A reduce transform is provided to take all values for a specific key, and generate a new list of the reduced output.
  - reduce(ey2, list<value2>) -> list<value3>
- MapReduce is the programming paradigm used by Hadoop for large data analysis. It basically applies the divide-and-conquer strategy within a distributed cluster of machines:
- MapReduce is a paradign used for large data analysis. It uses divide and conquer strategy in distributed clusters of machines it is generally divided into three parts :
  1. Split the original input data into many chunks (By default size of the block is 64 MB)
  2. This process is executed for each chunk in distributed manner and execute each process on the partial input data. It helps to generate partial output data is names as a mapper.
  3. Fe more processes are executed in charge of gathering the output of mappers and execute each process with aggregation or joining function. These processes are named as reducer.

Sample Code

Mapper Function :

```
import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
public class mapper extends Mapper<LongWritable, Text, Text, IntWritable> {

    @Override
    public void map(LongWritable key, Text value, Context context)
          throws IOException, InterruptedException {
       if (key.get() > -1) {
           // Convert each line of data to string for processing
           String line = value.toString();
           // Split up each element of the line
           String[] elements = line.split(" ");
```

```
                // news group name is second field
                String newsgroup = elements[0];
                context.write(new Text(newsgroup), new IntWritable(1));
            }
        }

}
```

Reducer Function :

```
import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;


public class reducer extends Reducer<Text, IntWritable, Text, IntWritable> {

    public void reduce(Text key, Iterable<IntWritable> values, Context context)
            throws IOException, InterruptedException {
        int count = 0;
        // Go through each element and count repetitions
        for (IntWritable value : values) {
            count++;
        }
        context.write(key, new IntWritable(count));

    }

}
```

Driver :

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
public class Driver {
    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        // Use programArgs array to retrieve program arguments.
        String[] programArgs = new GenericOptionsParser(conf, args)
                    .getRemainingArgs();
        Job job = new Job(conf);
        job.setJarByClass(Driver.class);
        job.setMapperClass(mapper.class);
        job.setReducerClass(reducer.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

        FileInputFormat.addInputPath(job, new Path(programArgs[0]));
        FileOutputFormat.setOutputPath(job, new Path(programArgs[1]));

        // Submit the job and wait for it to finish.
        job.waitForCompletion(true);
        // Submit and return immediately:
        // job.submit();
    }

}
```

- As shown in above code initially we have divided input files into the multiple chunks and it is given to the name node. To perform initial stage we have used Mapper function. Once we have done with mapper function we have started combining the multiple chunks to final out the final output. This process is carried out by reducers. Finally after passing the any text file to our program it will count number of words in the file and gives us a desire output.

2. Hadoop

Apache Hadoop is an open source framework for distributed storage and processing of large sets of data on commodity hardware. Hadoop enables businesses to quickly gain insight from massive amounts of structured and unstructured data. It is used in maintaining, scaling and analyzing large scale of data. This data can be structured or unstructured.

3. PIG

Apache PIG is a platform for analyzing large data sets. PIG‘s language, PIG Latin, lets one specify a sequence of transformation functions like merge, filter, grouping etc. Apart from built-in functions it also provides facility for user defined functions to do special-purpose processing. PIG‘s language allows for query execution over data stored on a Hadoop cluster, instead of a "SQL-like" language.

4. HIVE

Hive enables traditional BI applications to run queries against a Hadoop cluster. It was developed originally by Facebook, but has been made open source for some time now, and it's a higher-level abstraction of the Hadoop framework that allows anyone to make queries against data stored in a Hadoop cluster just as if they were manipulating a conventional data store. It makes Hadoop more useful for BI users.

5. Column oriented databases

Conventional, row-oriented databases are best fit for online transaction processing with high update speeds, but they fall short on query performance as the data volumes grow and as data becomes more unstructured. Column-oriented databases store data with a focus on columns, instead of rows, allowing for huge data compression and very fast querying.

6. Schema less databases, or NoSQL databases

There are several database types that fit into this category, such as key-value stores and document stores, which focus on the storage and retrieval of large volumes of unstructured, semi-structured, or even structured data. They achieve performance gains by doing away with some (or all) of the restrictions traditionally associated with conventional databases, such as read-write consistency, in exchange for scalability and distributed processing.

7. Using cloud for Big data

Most of the above technologies demand cloud, i.e ,many of the products and platforms mentioned are either entirely cloud-based or have cloud versions themselves. Most cloud vendors are already offering hosted Hadoop clusters that can be scaled on demand according to their user's needs. Big Data and cloud computing go hand-in-hand. Cloud computing enables companies of all sizes to get more value from their data by enabling faster analytics at minimal costs. This, in turn improves the company‘s productivity and thus returns the costs invested.

8. Oozie

Oozie is a workflow scheduler system to manage Apache Hadoop jobs. Oozie Workflow jobs are Directed Acyclical Graphs (DAGs) of actions. Oozie Coordinator jobs are recurrent Oozie Workflow jobs triggered by time (frequency) and data availability. Oozie is integrated with the rest of the Hadoop stack supporting several types of Hadoop jobs out of the box (such as Java map-reduce, Streaming map-reduce, Pig, Hive, Sqoop and Distcp) as well as system specific jobs (such as Java programs and shell scripts). Oozie is a scalable, reliable and extensible system.

9. Apache Spark

Apache spark is unified analytics engine used for large scale data processing. It is fast cluster computing system. It also provides a API for Scala, Python, R and Java. It is optimized engine that supports general execution graph. It supports higher level tools like Spark SQL for SQL and structured data processing, MLlib for Machine learning, Graphx for graph processing and spark streaming for streaming.

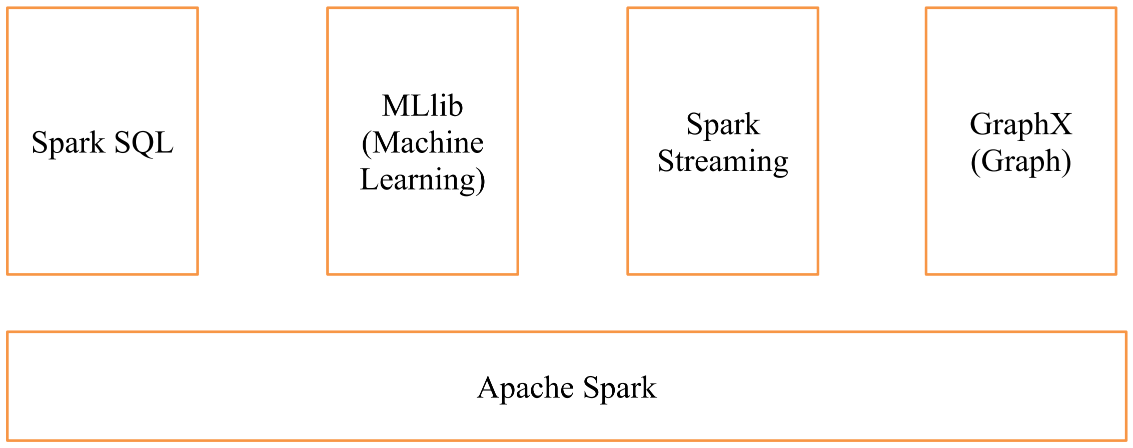


Fig. 6.5.1 : Apache spark framework

Characteristics of Apache Spark

1) **Speed :** It can achieve high performance for batch and streaming data. It uses DAG Scheduler, Query optimizer and physical execution engine to run workload 100x faster.
2) **Ease of Use :** Apache Spark provides more than 75 high level API to build parallel apps and that can be used interactively with Python, R, Scala SQL Shell.
3) **Generality :** It provides a solution to combine multiple libraries seamlessly in one application. It includes libraries like MLlib, SQL, Data frames and Grpahx.
4) **Runs Everywhere :** Spark can be run on cloud, hadoop, kuberenets, standalone systems. It can access diverse data sources.

6.6 Introduction to Hadoop

- Hadoop is open source java based framework used to store and process large amount of data.
- Hadoop framework is from Apache foundation and open source means code is available for free and user can also change the code according to its requirement. i.e. user wants to add certain functionality then he can add it.
- Hadoop allows you to run your tasks on multiple nodes of clusters. Parallel processing is achieved with the help of Hadoop.
- Basic programming language for hadoop is java, but you can code it in any languages like C, C++, Pearl and Ruby.
- Hadoop can be used with stand alone computer but to achieve parallel processing it should be used with cluster of machines.

- Hadoop has master – slave architecture. There is a one master node and number of slave nodes. Master node manages and monitors slave nodes.
- Slave nodes are responsible for performing actual task. Task is divided into number of tasks which is performed by this individual slave nodes and parallel processing is achieved.
- For performing actual task. Task is divided into number of tasks which is performed by this individual slave nodes and parallel processing is achieved.

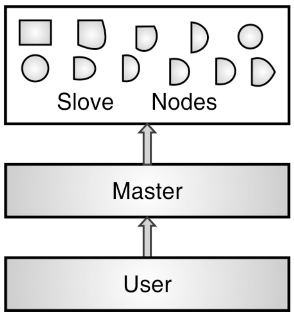


Fig. 6.6.1 : Master slave architecture for Hadoop

6.6.1 Pillars of Hadoop

Following are three Pillars of Hadoop :

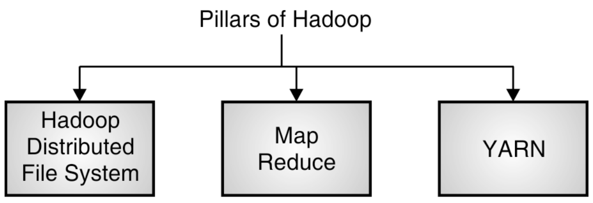


Fig. 6.6.2 : Pillars of Hadoop

6.6.1(A) Hadoop Distributed File System

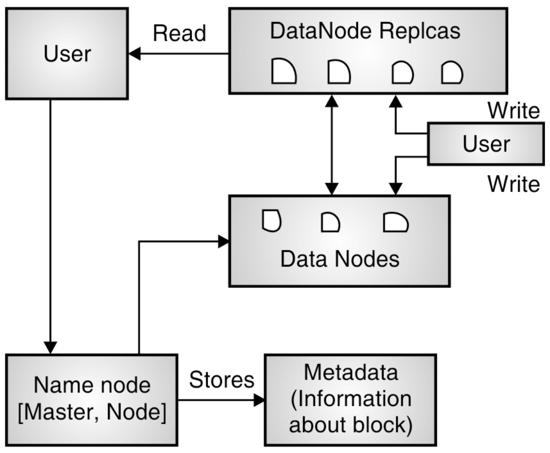


Fig. 6.6.3 : Hadoop distributed file system

- It is Pillar of hadoop which is used for achieving Fault Tolerance and Scalability.
- In HDFS, Master node is called as NameNode where Slave nodes are called as DataNodes.

- In this File is divided into number of blocks and is sent to different nodes i.e DataNodes.
- These blocks are again replicated to other nodes. So that failure of one node can be recovered using these replicas.
- NameNode has meta data (i.e. Data about data) regarding DataNodes like Block information, replica information, task given etc.

6.7 Data Analytics

Data Analytics is a most trending branch of computer science. Data analytics is a process of mining or extracting, loading, transforming, modeling and analyzing data and reach to one particular conclusion to make a decision. All this operations are happens at Database level. Once the data is formatted using above operations, it is converted into reports and these are Descriptive, Diagnostics, Predictive and Prescriptive Analytics.

6.7.1 Types of Data Analytics

- Data Analytics is used in various fields like Weather analysis, Agriculture, Industry etc.
- Predictive and Prescriptive analytics are use to work past data and envisioned it into future data. The below Fig. 6.7.1 shows four types of Data Analytics.

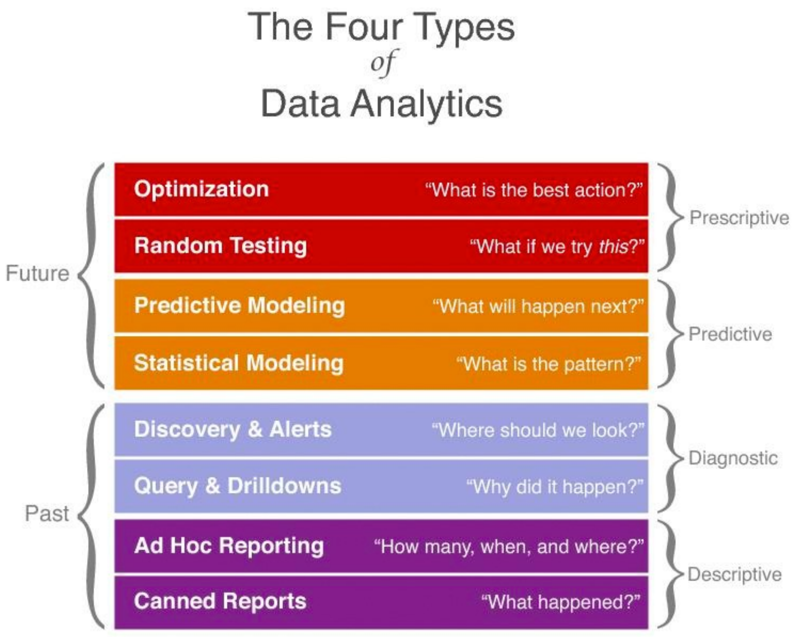

The Four Types of Data Analytics
Future
Past
Optimization "What is the best action?"
Random Testing "What if we try this?"
Predictive Modeling "What will happen next?"
Statistical Modeling "What is the pattern?"
Discovery & Alerts "Where should we look?"
Query & Drilldowns "Why did it happen?"
Ad Hoc Reporting "How many, when, and where?"
Canned Reports "What happened?"
Prescriptive
Predictive
Diagnostic
Descriptive

Fig. 6.7.1: Types of data analytics

Following are types of Data Analytics :

1. **Descriptive analytics :** This type use to reporting a bedrock. It is designed to get t basic Exploratory Analysis I that is, it Answers W5H questions. It answers What, when, why, where and how. This types design to extract the necessary information as per users choice. A developers can display or find out the current issues in this type. For e.g. If we consider the example of weather analysis in this case our system will answers the exploratory information

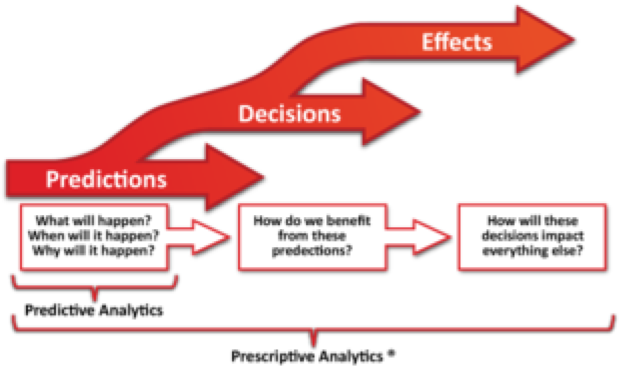
related to weather on the basis of available information. It will not predict any further information in Descriptive analytics.

2. **Diagnostics analytics :** It helps to answer the question Why something happens. It helps to find out the more detailed information and surprising values. For example, In the healthcare industry, customer segmentation coupled with several filters applied (like diagnoses and prescribed medications) allowed measuring the risk of hospitalization.
3. **Prescriptive analytics :** It is a third and main phase of analytics. Which includes descriptive and predictive Analytics. It entails the application of computational and mathematical and suggest the option to avail the benefits of descriptive and predictive analytics. In this type descriptive analytics looks at past experience and find the reason for failures or perform a exploration of data. It use in most management reporting like Finance, sales, marketing etc. As shown in above Fig. 6.7.1 prescriptive analytics is divided into 3 parts that is predictions, Decisions and Effects. There are many scientific streams which are comprises of prescriptive analytics. Such as machine learning, computer vision, operation research, natural language processing, image processing etc.
4. **Predictive analytics :** It is a type of analytics or area of statistics used for extracting the information from data and to. Predict the behavioral patterns and trends. The enhancement of web analytics can calculate possibilities of future online. It includes Machine Learning, behavioral modeling, deep learning and Artificial Intelligence algorithms. E.g. Identifying the suspects after crime happens or predict the weather conditions in advance.

Predictive analytics process is divided into seven parts

1. **Define project :** Define the project outcomes, deliverable, scope of the effort, business objectives, identify the data sets that are going to be used.
2. **Data collection** : Data mining for predictive analytics prepares data from multiple sources for analysis. This provides a complete view of customer interactions.

3. **Data analysis :** Data Analysis is the process of inspecting, cleaning and modeling data with the objective of discovering useful information, arriving at conclusion
4. **Statistics :** Statistical Analysis enables to validate the assumptions, hypothesis and test them using standard statistical models.


Effects
Decisions
Predictions
What will happen? When will it happen? Why will it happen?
How do we benefit from these predections?
How will these decisions impact everything else?
Predictive Analytics
Prescriptive Analytics *

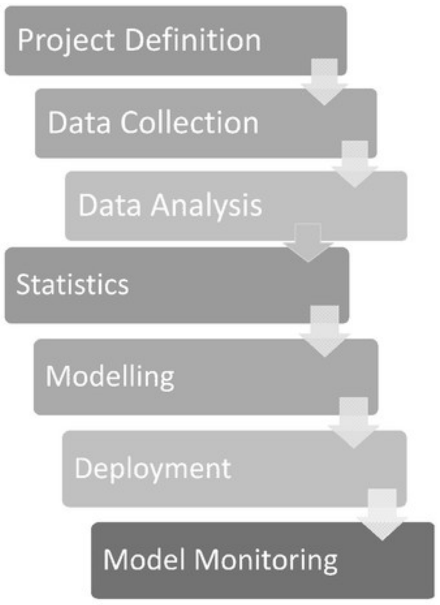


Fig. 6.7.2 : Stages of predictive analytics

5. **Modeling :** Predictive modeling provides the ability to automatically create accurate predictive models about future. There are also options to choose the best solution with multi-modal evaluation.
6. **Deployment :** Predictive model deployment provides the option to deploy the analytical results into everyday decision making process to get results, reports and output by automating the decisions based on the modeling.
7. **Model monitoring :** Models are managed and monitored to review the model performance to ensure that it is providing the results expected.

6.8 Apache Storm for Real-time Data Analysis

- Apache Storm is a free and open source distributed realtime computation system. Apache Storm makes it easy to reliably process unbounded streams of data, doing for realtime processing what Hadoop did for batch processing. Apache Storm is simple, can be used with any programming language, and is a lot of fun to use.
- Apache Storm has many use cases: realtime analytics, online machine learning, continuous computation, distributed RPC, ETL, and more. Apache Storm is fast: a benchmark clocked it at over **a million tuples processed per second per node**. It is scalable, fault-tolerant, guarantees your data will be processed, and is easy to set up and operate.
- Apache Storm integrates with the queuing and database technologies you already use. An Apache Storm topology consumes streams of data and processes those streams in arbitrarily complex ways, repartitioning the streams between each stage of the computation however needed.
- Analytics is about finding meaningful patterns in data. Real-time analytics is about building patterns by analyzing events as they occur. Traditional analytics is based on offline analysis of

historical data, whereas real-time analytics involves comparing current events with historical patterns in real time, to detect problems or opportunities.

- For example, in telecom when a subscriber uses a service on his/her terminal, it results in data exchange with his/her service provider. Typically, the telecom service provider just provides the requested service and stores charging data for billing / future reconciliation purpose.
- Here, real time analytics provides an opportunity to create value-added applications by processing these incoming data streams. These value-added applications can track the quality of service offered to its subscribers, in real time. The service provider can see how many of its subscribers are experiencing problems with data connectivity and call drops. In real time, the service provider can investigate the root cause of most of the problems.
- With few mouse-clicks, it can find out whether it is due to poor coverage or network congestion or a misfit service plan or it is a problem due to subscriber is terminal.
- This insight can help the service provider to take corrective actions in real time. The service provider can address the issues with coverage and congestion proactively, before complains start pouring in. Similarly, it can offer a more appropriate service plan or terminal to its affected subscribers, which can improve their overall experience.
- Big data technologies enable us to process data in large volumes as well as high velocity. Technologies like Hadoop makes it possible to run batch-oriented analytics over terabytes of data. Similarly, technologies like Apache Storm enable us to develop run-time analytics solutions which can process thousands of messages per second. Apache Storm helps us address the challenges of developing a reliable distributed solution such that our development effort is focused on business logic rather than implementing a distributed platform.

6.8.1 Business Challenges

- Traditionally, in an incoming service request we avoid additional processing. For example the processing which is required to look for patterns and statistics. Especially, when rate of the incoming data stream is of the order of thousands messages per second because any delay introduced by the additional processing can result in bottleneck in service delivery. Such delays can trickle down to impact on quality of service.
- For example, in telecom, a service provider may have complete details of each service taken by each of its subscribers. However, a service provider may have subscribers in order of 1,000,000s and analyzing service problems in real-time may require capability of processing 1,000s of messages per second (considering the aggregate traffic generated by these subscribers during a busy hour). Moreover, the system may also have to analyze performance data reported from the network infrastructure itself.
- To handle such a high rate of incoming data, solution would typically require distributed computing and high availability. Traditional solutions for building these clusters and managing them are expansive and require good amount of maintenance effort. Besides, a traditional

distributed solution has its own set of challenges like trading off between reliability and performance.

6.8.2 Traditional Approaches and its Disadvantages

- Traditionally, real time processing is done using queues and workers. For example, first there is a worker to receive messages and to place it in a queue. Second worker listens to the queue, picks up the message, parses it, processes it and places the processed message in a second queue. Next worker picks up the processed message from the second queue and logs it in database.

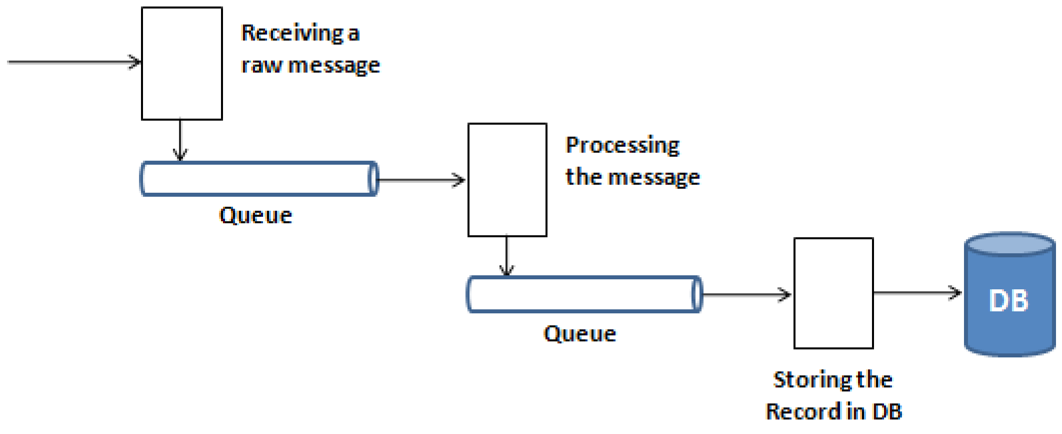


Fig. 6.8.1 : Traditional approaches

- This approach has several disadvantages. First, it needs to be made sure that all queues and workers stay up, all the time. Secondly, when application is required to scale to support high throughput, then these queues and workers are required to be distributed. Distributing the queues, requires exchange of state across all the nodes within the cluster and there is a trade-off between reliability and performance of the distributed queues.
- Distributed queues and fault-tolerance are complex functions. Implementing life cycle management of the queues (setting up, monitoring, cleanup, setting-up again) involves significant development time.
- Especially, fault tolerance and high-availability require good amount of development time and their maintenance is also quite expensive. Usually, these functions cannot be tested in lab environment because problems appear only in high load conditions.
- Moreover, when these problems occur in field, then one does not get debug-traces for troubleshooting. In short, it will be a great relief if an off-the-shelf framework can provide distribution and fault tolerance along with high performance, for real time processing.

6.8.3 Apache Storm

- Apache storm is a framework that facilitates distributed real-time processing. Real-time processing of a data stream with Storm, is like water treatment of a stream of water.

- The way stream of water is taken through various stages, different procedures are performed in each stage and output of one stage is fed as input to the next stage.
- Similarly, during real-time processing of a stream of data by Storm, a spout reads the incoming data stream and feeds it to a processing unit called bolt. Bolt performs specified business logic on the data stream and passes on the processed stream to the next bolt. Bolts and interconnections between them are termed as storm topology. There can be one or more topologies in a single system, which can be connected through message queues.

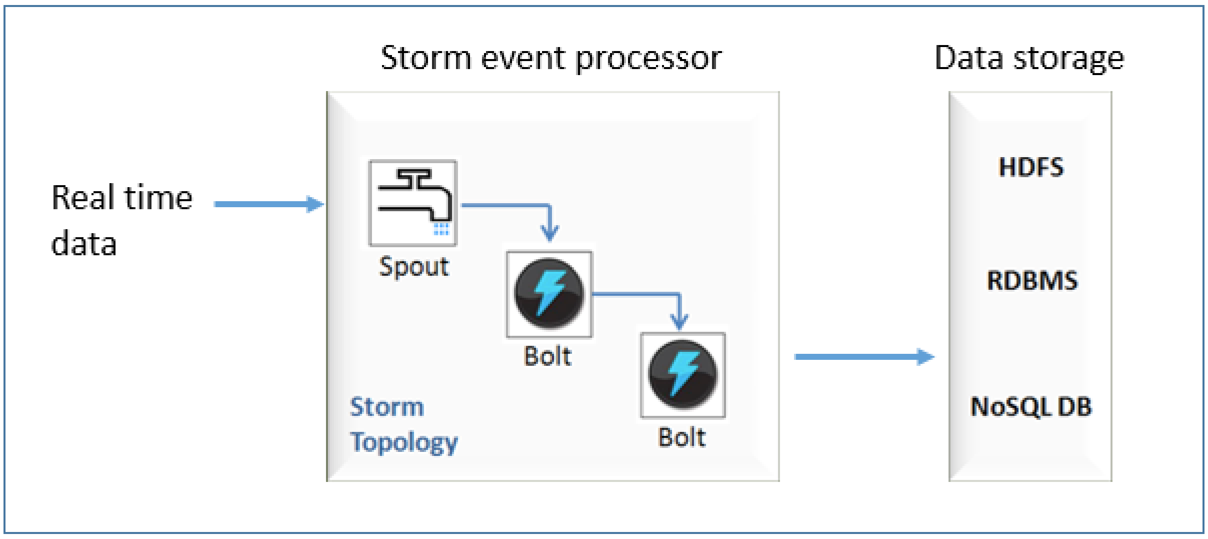


Fig. 6.8.2 : Apache Storm

- Each spout and bolt can have one or more instances. These instances can be spread throughout the storm cluster. Communication among these distributed spouts and bolts is transparent to a developer.
- The communication is seamlessly managed by the storm framework itself. Storm is open-source and is available free-of-cost (https://storm.apache.org/about/free-and-open-source.html). To use Storm, a developer is primarily required to provide implementation of execute() method of each bolt. Following features come out-of-the-box with Apache Storm :

1. Distributed framework (scalable)
2. Fault tolerant (highly available)
3. Guarantees no data loss (reliable)

- Storm topology can be easily integrated with different data storage options, like HDFS, traditional RDBMS, and a NoSQL database.

6.8.4 Real-Time Analytics With Network Data

- This section explains Apache Storm based real-time analytics solution, using an example of a telecom service provider. In the network of a telecom service provider, there can be different sources of incoming data, like :
  - Stream of data generated due to use of services by subscribers.

- Performance data of access network, as reported by network probes.
- Data related with new subscription orders, activation and terminate orders.

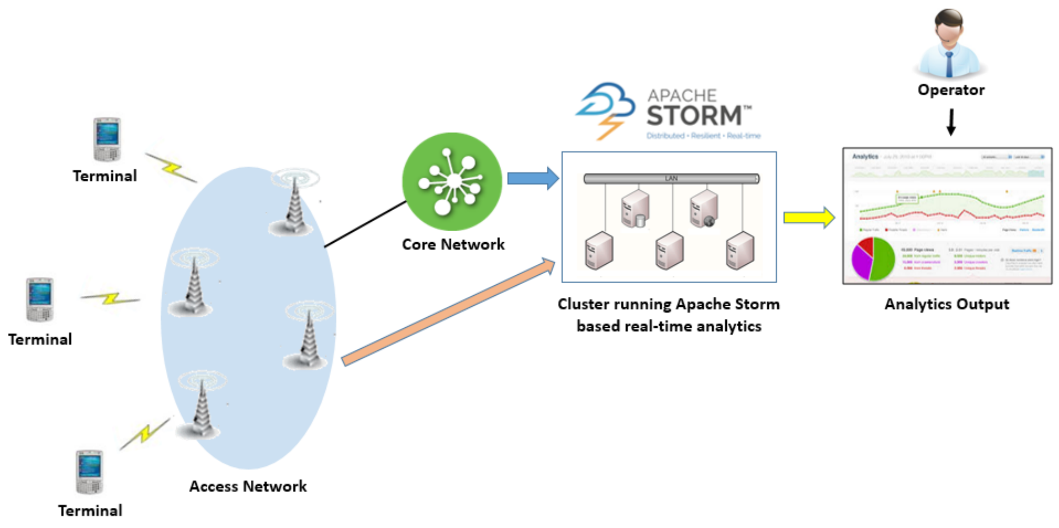

Terminal
Terminal
Terminal
Access Network
Core Network
APACHE STORM
Distributed • Resilient • Real-time
LAN
Cluster running Apache Storm based real-time analytics
Operator
Analytics Output

Fig. 6.8.3 : Real-time analytics

- A single storm topology running inside a cluster can listen to all different kinds of incoming data. The storm topology may have different spouts for each source of input. Spouts are followed by Bolts which take care of processing the incoming data and calculate statistics which captures information across services, regarding the quality of service provided to the subscribers, performance of network elements and new received orders. The processed statistics are stored in a NoSQL database for future reference and are also used to plot several services-related real-time charts.
- The analytics can be used to device a solution that can automatically detect under-performing network elements and reconfigure the network such that it is more balanced with respect to its utilization. Similarly, network coverage related issues can be tracked and input can be provided to network planning, in real-time. Also, the analytics charts can be monitored by proactive call center operators, who can reach out to subscribers (who are facing service related issues) with attractive offers to replace their service plan or terminal which suits better to their usage pattern.
- Such seamless handling of network-related issues and proactive customer support can be instrumental in improving user experience manifold. It can help a telecom service provider in building a competitive advantage in the market, by maintaining a real-time focus on customer experience.
- Apache Storm provides a stable and robust framework for a real-time analytics solution. The framework provides base classes for spouts and bolts. Spout class inherits class BaseRichSpout and bolt class inherits BaseRichBolt.

- One is required to just implement nextTuple() method in spout class such that it reads data from an incoming data stream and emits it inside the storm topology.
- Similarly, one has to write the implementation of execute() method in bolt class to provide business logic to process the data passed on by the connected spout. Multiple spouts can be defined for different sources of data.
- For example, one spout for tapping into charging data, second to tap performance data from the access network and third spout for accessing data from incoming order requests.
- In short, it is a collection of spouts and bolts (interconnected with each other) which is referred to as storm topology. Business logic to process the incoming stream of data is embedded within this topology.
- Storm framework distributes this topology across different nodes of the cluster and ensures reliability along with high throughput (off-the-shelf). This way development remains focused on the business logic of the data processing.

6.9 Tools for IoT

6.9.1 Chef

- Chef Infra is a powerful automation platform that transforms infrastructure into code. Whether you're operating in the cloud, on-premises, or in a hybrid environment, Chef Infra automates how infrastructure is configured, deployed, and managed across your network, no matter its size.
- This Fig. 6.9.1 shows how you develop, test, and deploy your Chef Infra code.

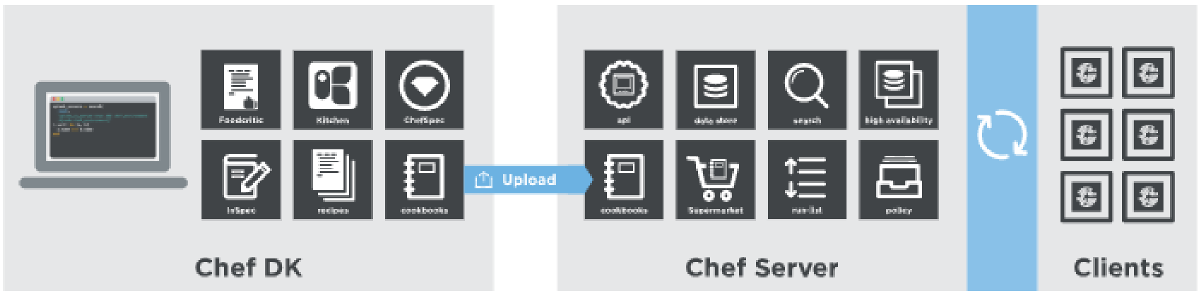


Fig. 6.9.1 : Chef

- Chef Workstation is the location where users interact with Chef. With Chef Workstation, users can author and test cookbooks using tools such as Test Kitchen and interact with the Chef Infra Server using the Knife and chef command line tools.
- **Chef Infra Client nodes** are the machines that are managed by Chef. The Chef Infra Client is installed on each node and is used to configure the node to its desired state.

- **Chef Infra Server** acts as <u>a hub for configuration data</u>. Chef Infra Server stores cookbooks, the policies that are applied to nodes, and metadata that describes each registered node that is being managed by Chef. Nodes use the Chef Infra Client to ask the Chef Infra Server for configuration details, such as recipes, templates, and file distributions.

6.9.2 Puppet

Puppet is a Configuration Management tool that is used for deploying, configuring and managing servers. It performs the following functions :

- o Defining distinct configurations for each and every host, and continuously checking and confirming whether the required configuration is in place and is not altered (if altered Puppet will revert back to the required configuration) on the host.
- o Dynamic scaling-up and scaling-down of machines.
- o Providing control over all your configured machines, so a centralized (master-server or repo-based) change gets propagated to all, automatically.

- Puppet uses a Master Slave architecture in which the Master and Slave communicate through a secure encrypted channel with the help of SSL.

Puppet Case Study

- System Administrators usually perform repetitive tasks such as installing servers, configuring those servers, etc. They can automate this task, by writing scripts, but it is a very hectic job when you are working on a large infrastructure.
- To solve this problem, *Configuration Management* was introduced. Configuration Management is the practice of handling changes systematically so that a system maintains its integrity over time. Configuration Management (CM) ensures that the current design and build state of the system is known, good & trusted; and doesn't rely on the tacit knowledge of the development team. It allows access to an accurate historical record of system state for project management and audit purposes. Configuration Management overcame the following challenges:
  - o Figuring out which components to change when requirements change.
  - o Redoing an implementation because the requirements have changed since the last implementation.
  - o Reverting to a previous version of the component if you have replaced with a new but flawed version.
  - o Replacing the wrong component because you couldn't accurately determine which component needed replacing.

Let us understand its importance through a use case.

- The best example I know is of the New York Stock Exchange (NYSE). A software "glitch" prevented the NYSE from trading stocks for almost 90 minutes. This led to millions of dollars of loss. A new software installation caused the problem. That software was installed on 8 of its 20 trading terminals and the system was tested out the night before. However, in the morning, it failed to operate properly on the 8 terminals.

- So there was a need to switch back to the old software. You might think that this was a failure of NYSE's Configuration Management process, but in reality it was a success.
- As a result of a proper Configuration Management process, NYSE recovered from that situation in 90 minutes which was pretty fast. Had the problem continued longer, the consequences would have been more severe.

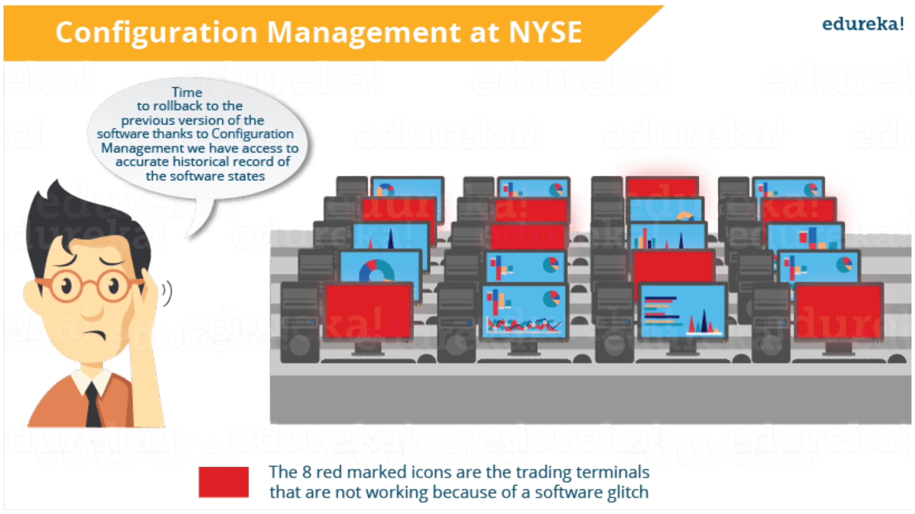

Configuration Management at NYSE
edureka!
Time to rollback to the previous version of the software thanks to Configuration Management we have access to accurate historical record of the software states
🟥 The 8 red marked icons are the trading terminals that are not working because of a software glitch

Fig. 6.9.2

- Now, I hope you know the importance of Configuration Management. Configuration Management stage can be considered as the backbone of DevOps. It allows more frequent software releases in the safest and most reliable way possible.

6.9.3 NETCONF and YANG

- The IETF's Network Configuration Protocol and Yang data modeling language promise to help simplify and speed network device configuration.
- This vendor-written tech primer has been edited by Network World to eliminate product promotion, but readers should note it will likely favor the submitter's approach.
- Service provider and enterprise network teams that are moving towards a service oriented approach to network management are reaching for the IETF's Network Configuration Protocol (NETCONF) and YANG, a data modeling language, to help remove the time, cost and manual steps involved in network element configuration.
- NETCONF is the standard for installing, manipulating and deleting configuration of network devices while YANG is used to model both configuration and state data of network elements. YANG structures the data definitions into tree structures and provides many modeling features, including an extensible type system, formal separation of state and configuration data and a variety of syntactic and semantic constraints. YANG data definitions are contained in modules and provide a strong set of features for extensibility and reuse.

- What does this mean? Network automation is currently blocked by current approaches where you need to write device specific CLI scripts or are locked into rigid closed tools. There is nothing wrong with CLIs; they are perfect for humans, but less optimal for software. NETCONF is defined for transaction-safe configuration of devices. This means that scenarios like setting up initial configuration for a range of devices, changing ACLs and adding VPNs, can be performed automatically, while keeping flexibility and vendor independence.
- Additionally, time-to-market requirements in delivering new services are critical and any delay in configuring the corresponding devices directly affects deployment and can have a big impact on revenue. Organizations are seeing the need to get the people out of the way to automate the configuration and implementation of network devices.
- Ultimately the technology is designed to support more robust management of configuration, including configuration change transactions including rollbacks, strong separation between configuration and state data allowing for efficient backup-and-restore, selective data retrieval with filtering using well-known query mechanisms and streaming and playback of event notifications.
- The current scale of networking across service providers, enterprises and cloud providers poses unprecedented challenges to operations teams. As both frequency and complexity of changes made to the network, as well as the cost of failed configurations explode, network operations teams understand the cost savings that come with delivering services quickly and are now requiring the use of NETCONF and YANG in their environments to achieve these benefits. Compound this with all the other challenges organizations face including frequent network changes, service agility, network complexity, SLAs becoming tighter and simply doing more with less, it is no wonder networking teams are losing sleep.
- With the network industry being quite conservative; there are a couple of things that need to happen for any management technology to take hold:
  - First, the technology needs to be implemented in mainstream network products. This means that support for the NETCONF protocol needs to be included with versions of router and switch operating systems like JunOS from Juniper and IOS-XR from Cisco.
  - Second, there needs to be support for the technology in software used by operations teams that run the network. This may include command line tools for network engineers that can troubleshoot issues all the way up to the top while supporting the provisioning and orchestration parts of the OSS/BSS stack.
  - Finally, and most importantly – the main driver of implementation and adoption for both items is directly related to whether end-users are asking for it. Unless technologies like NETCONF are explicitly named in RFIs and RFQs, there is little incentive for the vendors to implement it.

6.9.4 IoT Code Gernerator

The heterogeneous compiler is the main difference between the traditional dataflow applications optimized for high-speed computing and the IoT applications. Heterogeneous dataflow has been a major research work but prior research mainly concentrates on multiple processing cores (of same type like multicore or of different types as CPU-GPU) integrated into the same computing hardware, often on the same chip.

IoT applications are inherently heterogeneous and distributed with communication links that can be unreliable (like short-range communication links) and/or slow (like low-power WANs).

The following Figure demonstrates, how the heterogeneous code generator backend fits into the Orcc architecture. The heterogeneous backend knows, which parts of the dataflow application are assigned to which homogeneous backend, partitions the graph and hands parts of the graph to their associated homogeneous code generators.

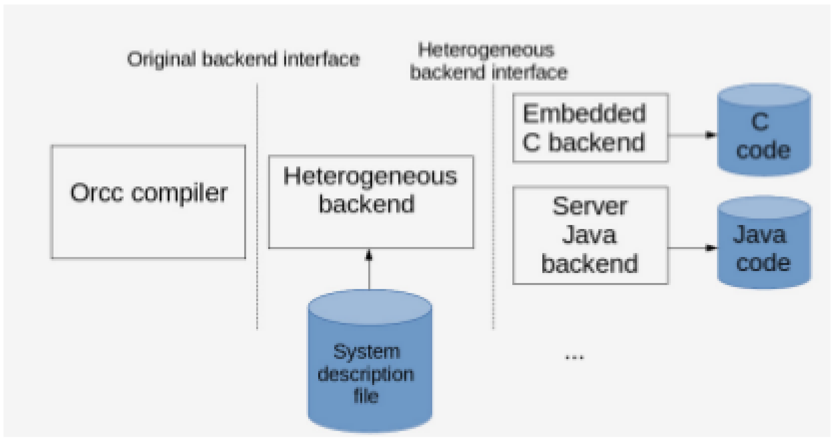


Fig. Heterogeneous code generator architecture

6.9.5 Chef Tutorial – Chef Server

The Chef Server acts as a hub for configuration data. The Chef Server stores Cookbooks, the policies that are applied to Nodes, and metadata that describes each registered Node that is being managed by the Chef-Client.

Nodes use the Chef-Client to ask the Chef Server for configuration details, such as Recipes, Templates, and file distributions. The Chef-Client then does as much of the configuration work as possible on the Nodes themselves (and not on the Chef Server). Each Node has a Chef Client software installed, which will pull down the configuration from the central Chef Server that are applicable to that Node. This scalable approach distributes the configuration effort throughout the organization.

Chef Tutorial – Chef Nodes

Nodes can be a cloud based virtual server or a physical server in your own data center, that is managed using central Chef Server. The main component that needs to be present on the Node is an agent that will establish communication with the central Chef Server. This is called Chef Client.

Chef Client performs the following functions:

- It is responsible for interacting with the central Chef Server.
- It manages the initial registration of the Node to the central Chef Server.

- It pulls down Cookbooks, and applies them on the Node, to configure it.
- Periodic polling of the central Chef Server to fetch new configuration items, if any.

Interoperability testing of devices

- Looking back at what we learned with SNMP, one of the challenges around standards is get beyond simply ticking off compliance in an RFP and actually supporting the standard according to the RFCs. In many cases users of technology suffer from non-conformant implementations. If vendors just put their CLI config within NETCONF tags nothing is gained.
- For NETCONF and YANG, this has happened in a slightly different order. Some of the equipment providers (e.g. Cisco, Juniper, Ericsson) showed the way by providing strategic early implementations of NETCONF (in contrast to customer-driven).
- The next second step in the process is happening right now. Network operations teams are at the point where manual configuration is a non-starter, so there is an urgent need for automation solutions. This is currently driving increased mentions of NETCONF and YANG in RFIs and RFQs. This in turn is drawing quite significant attention from the network management software companies that wants to stay relevant in these areas.