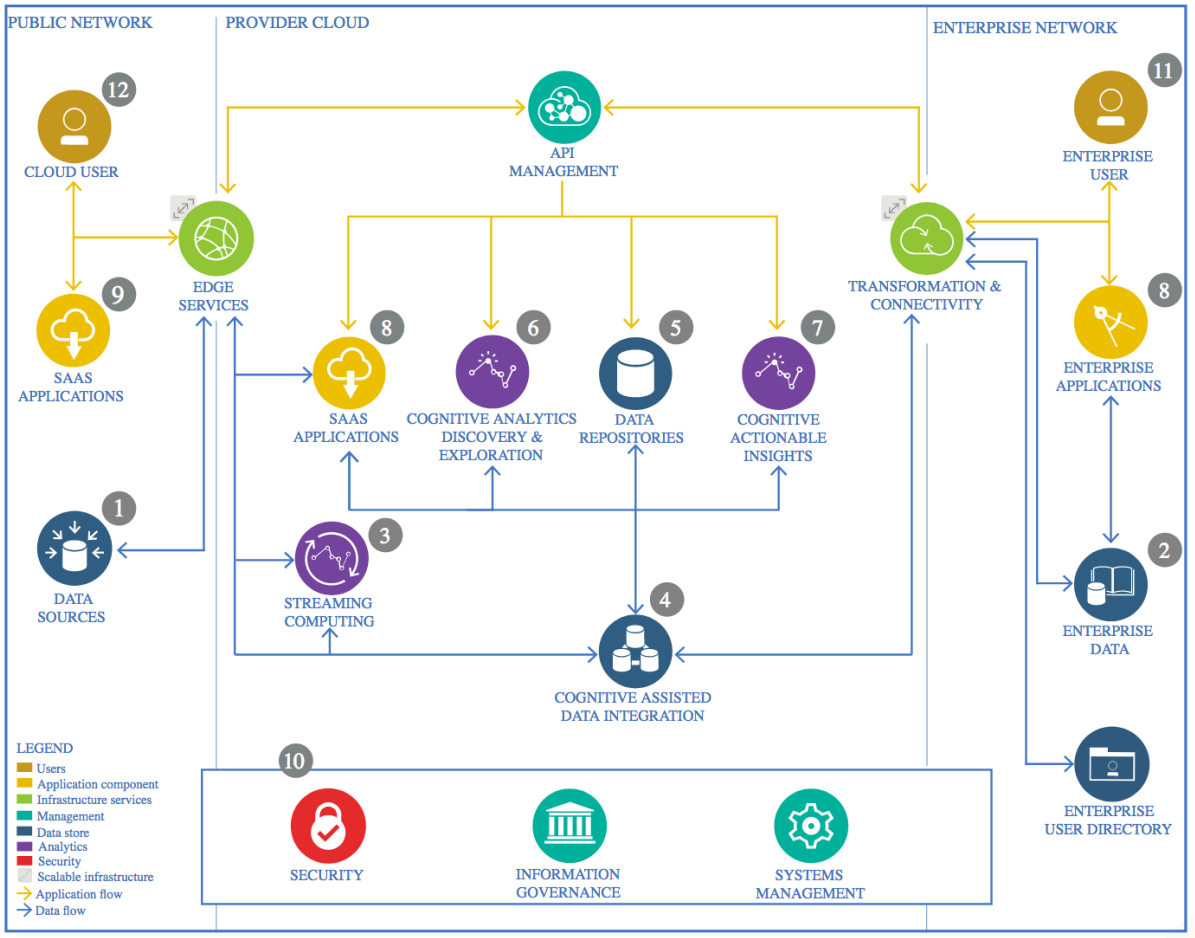
The Lightweight IBM Cloud Garage Method for Data Science

Architectural Decisions Document Template

1. Architectural Components Overview



IBM Data and Analytics Reference Architecture. Source: IBM Corporation

1.1. Data Source

1.1.1. Technology Choice

With data source, there is not much to decide as the majority of the type and structure of the data source is already defined by the stakeholders. However, the data scientist can advise against running SQL queries, part of ETL processes, against OLTP systems which will reduce performance. One solution is to use IBM Db2 Workload Manager because it allows workloads such as OLAP and ETL to run simultaneously without performance reduction using intelligent scheduling and prioritising mechanisms.

1.1.2. Real-time Data Source

Data can come in different shapes and delivery means but we just need a simple REST HTTP endpoint to be polled, a TCP or UDP one. If there are no real-time requirements, the data can be staged using Cloud Object Store.

1.2. Enterprise Data

1.2.1. Technology Choice

It is costly to move enterprise data to the cloud and hence, enterprises should only consider it when necessary. I'll recommend using Lift, a free and secure service hosted on IBM Cloud.

1.2.2. Justification

With IBM Lift CLI, it is simple to migrate data to a cloud property. It enables ultra-high-speed data transfer to the cloud via IBM Aspera, a highly-efficient data transport technology. Life automatically recovers from common problems faced during migration processes such as connection timeouts. Also, the connection to the IBM Cloud is secured via a 256-encrypted connection.

1.3. Streaming analytics

1.3.1. Technology Choice

Apache Spark is often the main choice when it comes to data processing and machine learning. Without further ado, Apache Spark is recommended.

1.3.2. Justification

Apache Spark Structured Streaming supports simultaneous operation of batch and streaming jobs to achieve a much higher throughput. With the introduction of the Continuous Processing mode, the latency has been brought down to one millisecond. Apache Spark performs well with structured and semi-structured data but audio and video datasets does not benefit from Apache Spark's accelerators such as Tungsten and Catalyst. Apache Spark Structured Streaming is able to perform relational queries as well as machine learning and through their fault tolerant nature, clusters can grow and shrink dynamically. Apache Spark Structured

Streaming guarantees delivery and depending on the type of data source, a complete crash fault tolerance can be achieved.

1.4. Data Integration

1.4.1. Technology Choice

Apache Spark is the first choice when it comes to cluster-grade data processing and machine learning. It is also a flexible option that support writing integration processes in SQL and hence, Apache Spark is recommended.

1.4.2. Justification

Apache Spark scales linearly with the cluster size and the throughput can be increased with the increase in cluster size. Apache Spark can access SQL and noSQL data as well as file sources. Having a common data source architecture makes it easy to add capabilities and third-party project functions. Hence, advanced SQL skills are required as well as either Java, Scale or Python.

1.5. Data Repository

1.5.1. Technology Choice

Since we are using Apache Spark for Streaming analytics and Data Integration, I would recommend Apache CouchDB, a noSQL database for file storage.

1.5.2. Justification

NoSQL databases are typically fault-tolerant and hence, quality requirements are less which brings down the cost of storage. NoSQL databases uses JSON as the storage format which can with enriched with binary data. With Apache CouchDB, it supports indexing which improves point and range query performance. In general, NoSQL performs very well at full table scans and the performance is only limited by the available bandwidth. Typically, special query language skills are required for the application developer. NoSQL databases are crash fault tolerant but for recovery, the system might need to go offline. Scaling of Apache CouchDB is also not an issue, volumes can be added at run time but for shrinking, the system might need to be taken offline.

1.6. Discovery and Exploration

1.6.1. Technology Choice

IBM Cloud has many offerings but I would recommend the open source ones such as Jupyter, Python, pyspark, scikit-learn, pandas, Matplotlib and PixieDust.

1.6.2. Justification

Matplotlib supports various data visualisations such as run charts, box-plots, scatter plots and histograms whereas PixieDust support tables, maps, histograms, pie charts, scatter plots, line and bar charts. With PixieDust, we can make interactive visualizations and coding skills are not needed. Using pandas and scikit-

klearn, all state-of-the-art metrics are supported. And with Watson Studio, it supports sharing of Jupyter Notebooks and also a more fine-grained user access management system.

1.6.3. Data Quality Assessment

From the initial data exploration, the dataset is very imbalanced as most transactions are non fraudulent. In order to avoid the problem of overfitting to the models, I have attempted Random Under Sampling. Hence, the amount of fraudulent and non-fraudulent transactions are reduced to the same amount of 492.

1.6.4. Feature Engineering

In this section, I attempt to perform feature engineering by filtering. As the input values sometimes does not correlate well with the output/classification, removing the input values/columns will be a better strategy to improve the classification accuracy.

1.7. Actionable Insights

1.7.1. Technology Choice

Although R and R-Studio have been open source for awhile, Python, pandas and scikit-learn are right behind them. Python is also a cleaner programming language so it is easier to read and learn. Pandas is the Python equivalent to R data frames also supporting relational access to data. Scikit-learn nicely groups all machine learning algorithms together which is nice. And it is all supported in IBM Cloud through IBM Watson Studio at no cost.

1.7.2. Justification

It is easy to find someone with Python skills as it is widely available and Python is a clean and easy to learn programming language and the cost of Python programming skills is very low. And for Pandas and scikit-learn, learning resources are widely available and the frameworks are very clean and easy to learn as well. All scikit-learn models can be serialised with PMML supported through various third-party libraries. Scikit-learn machine learning algorithms are cleanly implemented. As they all stick to the standard pipelines API, it is easy to reuse and interchange different pipelines. Linear algebra is handled throughout using the Numpy library which makes tweaking algorithms straightforward.

1.7.3. Algorithm Selection & Performance Indicator

For performing the initial data exploration, it is clear that we'll need clustering algorithms to work with the dataset and it is a binary classification problem. In this case, I have decided to go with K-Means clustering and Logistic Regression. First reason is to keep it simple and second, I want to use clustering algorithms that are covered in the previous modules in this specialisation, to put them to use. For the performance indicators, I'll go with Jaccard index and the F1-score and the LogLoss for the case of Logistic Regression.

1.8. Applications / Data Products

1.8.1. Technology Choice

I would recommend Node-RED, a no-code data integration integration environment. With its modular nature, it supports various extensions such as dash boarding extension. This allows fast creation of user interfaces including advanced visualisation that are updated real-time.

1.8.2. Justification

Due to the completely graphical user interface, only basic programming skills are required to build data products with Node-RED. But basic JavaScript knowledge is required for creating advanced flows and also when it comes to extending the framework. The UI provides real-time update of the data model. Therefore, all considerations regarding feedback delay need to be considered when developing data integration flow or potentially involved calls to synchronous or asynchronous 3rd-party services. The Node-RED dashboard can be deployed for a public user base as long as limitations regarding UI customisation of the dashboard is acceptable.

1.9. Security, Information Governance and Systems Management

1.9.1. Technology Choice

Object Storage is a standard when it comes to modern, cost-effective cloud storage.

1.9.2. Justification

IBM Identity and Access Management, IAM in short, integration allows for granular access control at the bucket-level using role-based policies. For data retention, it supports different storage classes for occasionally accessed data, frequently accessed data and long-term data retention with standard and cold vault. The Flex class allows for dynamic data access and automates this process. In terms of the operational aspects of Object Storage, regional and cross-regional resiliency options allow for increased data reliability. In terms of security needs, all data is encrypted at rest and in flight by defaults. Encryption keys are automatically managed by default but optionally, can be self-managed or managed using IBM Key Protect.