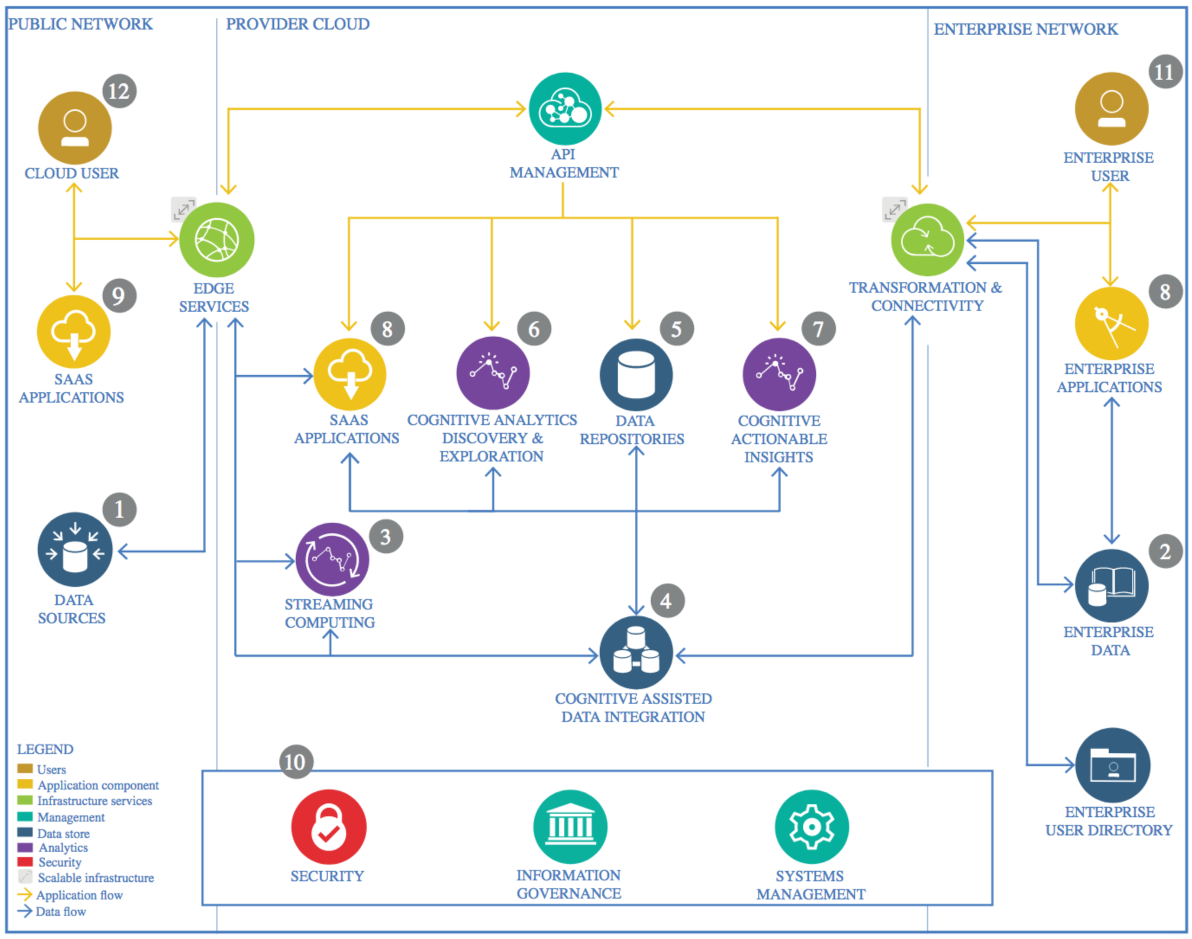
The Lightweight IBM Cloud Garage Method for Data Science

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

IBM gathers 3-4 million news articles each month through services such as webHose into an internal repository called CogNet. As the articles are ingested into the repository, the title and body are run against an NLU process to create various tags. I will be extracting data and tags from this repository for analysis and to build a classifier model.

1.1.2 Justification
The NLU process is costly in terms of cost, time and resources. If a classifier can be trained on existing tagged articles, it would be possible to tag articles at a lower cost to the organization.

1.2 Enterprise Data

1.2.1 Technology Choice
The initial data extract will be accomplished using Python and an elastic search. The result will be uploaded to Watson Studio as a CSV file.

1.2.2 Justification
Watson Studio does not connect well to the article repository (CogNet.) As such the initial data mining work will be done locally with Jupyter Notebooks.

1.3 Streaming analytics

1.3.1 Technology Choice
No streaming analytics will be used in the processing.

1.3.2 Justification
The news articles are already received from WebHose and other providers and processed into the CogNet repository.

1.4 Data Integration

1.4.1 Technology Choice
Step 1 – Data extraction: Python code using an Elastic Search against the Cogent data for article retrieval
Step 2 – Python code to strip any HTML tags and certain punctuation. Dump to a CSV file
Step 3 – Import the CSV file into the local object storage in Watson Studio
Step 4 – Use Watson studio (with Python) to build a classifier model

1.4.2 Justification
The choice of Python is one of convenience. Jupyter Notebooks are a common method for code development in Watson Studio.

1.5 Data Repository

1.5.1 Technology Choice
Moving from the data gathering phase to data exploration and modeling will require a transition from Jupyter Notebooks running locally to Watson Studio. To facilitate the transition the retrieved articles, titles and primary tag will be saved to a CSV file and uploaded to the local Watson Studio object store.

1.5.2 Justification
CSV is the 'lowest common denominator' and carries the lowest overhead. I expect to migrate about 30,000 news articles.

1.6 Discovery and Exploration

1.6.1 Technology Choice
After the articles are brought into the model I will need to identify some core characteristics of the data. As this will be a supervised learning model I want to identify the articles with the most and the least primary tags. Looking at the articles with the least tags I will need to cut off those tags without sufficient articles as there will not be enough data for the training of the model. I will start with a simple visualization, then a crosstab to count the occurrences.

1.6.2 Justification
Providing visualizations will help understand the data and bring the data to a point where the model can be trained.

1.7 Actionable Insights

1.7.1 Technology Choice
Python charts and crosstabs will be used.

1.7.2 Justification
The data consists of only three fields (Title, Body, Tag) there is not much more we can do here.

1.8 Applications / Data Products

1.8.1 Technology Choice
All work will be done in Jupyter Notebooks within Watson Studio utilizing Python

1.8.2 Justification
Python is the language of choice and the one I am most comfortable utilizing for this project.

1.9 Security, Information Governance and Systems Management

1.9.1 Technology Choice
No PII or other private information is involved in this work.

1.9.2 Justification
Not applicable.

2 Model

2.1 Model definition

2.1.1 Model choice

Several models will be built for this work. As this is a supervised model I will evaluate classifier results from:

- Naïve Bayes
- Logistic Regression
- Neural Network

2.1.2 Model Evaluation

For all models I will utilize the following evaluation:

- Precision
- Recall
- F1

The evaluation will be at the individual classifier and the overall model levels.

2.2 Model training

The data will be split into 70/30 training and testing splits for all three models. Given the large number of articles in use the 30% testing split will provide ample data.

In the Logistic Regression Classifier model:

- Split the data into train (70%) and test (30%)
- Tokenize the test and training data
- Use word averaging on the list
- Training will cover 30 epochs

The Neural Network Classifier model:

- Split the data into train (70%) and test (30%)
- Tokenize the test and training data
- Evaluate the results with batch sizes of 16,32 and 64
- Evaluate the results with 2,4,6,8,12 epochs
- Evaluate the model with differing numbers of layers and dropouts

2.3 Model deployment

2.3.1 Initial deployment

Feedback on the model will be provided as a PDF showing the generated insights and how the model was built.

2.3.2 Justification

For the initial model deployment this deliverable will not require further effort as the model assessment must be justified before additional work is done.

2.3.3 Future options

The long term goal will be a REST API allowing the intake of an article and returning the classification.