adatis

High Level Design

Next Generation Business Intelligence Platform

anglianwater

Version: 1.2

26 March 2018

Reference Information

Document Author(s):	Scott Hudson - Adatis
Date	21/02/2018
Document Version	1.2 - Following Final Review
Document Classification	Client Confidential
Distribution	

Document Approvals

Team / Group	Name	Date
Design Authority		

Contact Information

Address

Adatis Consulting Ltd
Broadmede House
Weydon Lane
Farnham
Surrey
GU9 8QT
United Kingdom

Contact

Scott Hudson
scott.hudson@adatis.co.uk
+44 7807886553
+44 1252 267 777

Copyright

Table of Contents

1 Executive Summary

1.1 Document Purpose

This document describes the architectural design for the Next Generation Business Intelligence (NGBI) Platform solution to provide a modern data warehouse architecture for Anglian Water Services (AWS) that meets their current and future analytical and reporting needs. It does not provide a design for the individual data marts/applications that will be built using the NGBI platform.

1.1.1 Scope

This document is a high-level design based on the Foundations PoC of the NGBI programme.

The scope of the NGBI platform is provided in section 1.3 of this document and section 1.4 sets out the parameters of the proof of concept.

The solution in scope for this document is:

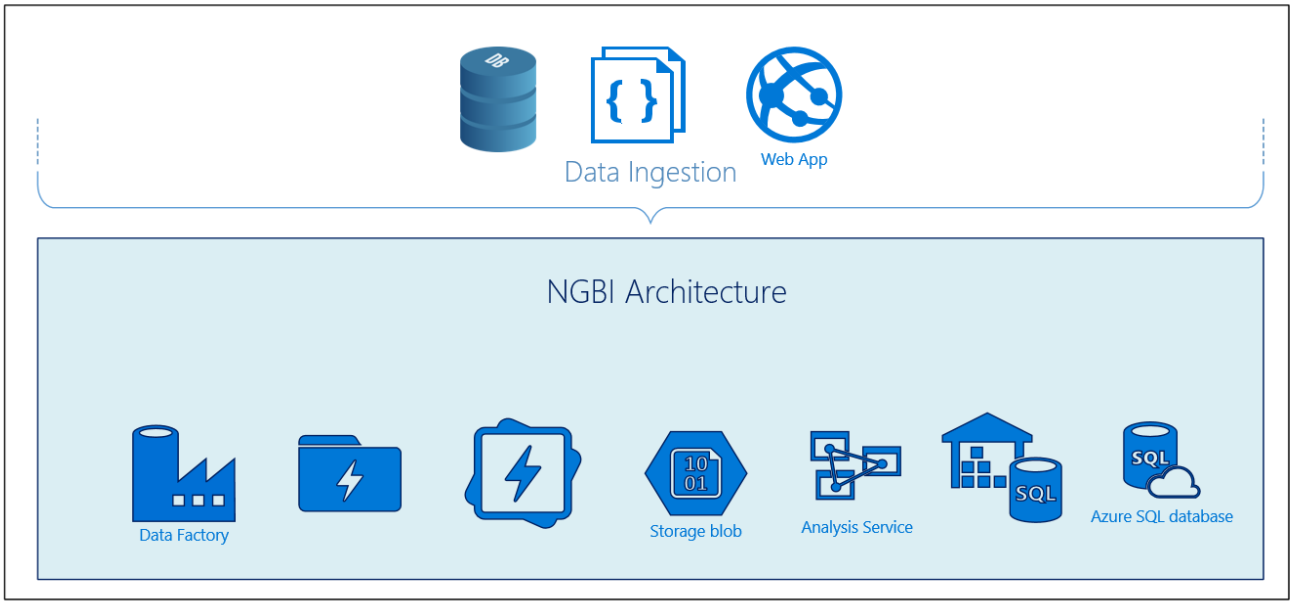
- Modern BI Architecture (Section 2.1.1)
- Applications Integration Architecture (2.1.2)
- Data Science and Exploration Architecture (2.1.3)

The NGBI Programme Solution Overview document will describe how the various workstreams and design documentation produced by the Programme will answer the question of how the overall solution meets business requirements and to what extent.

There are several architecture components which are described at a general conceptual level in this document, however, specific Anglian Water considerations will be documented during the subsequent detailed design phase. These include:

- Hybrid interim architecture (section 2.5) Please note that the migration, re-engineering or decommissioning of the interim platform will be described in subsequent documentation.
- Power BI architecture (section 3.7 and 4.7). Please note that subsequent documentation will define the approach to the enterprise deployment of Power BI.
- Other self-service reporting tools (apart from Power BI) (section 4.7.7). Please note that there will be subsequent documentation of the reporting semantic layer that will include describe how other self-service tools will interact.
- General Data Protection Regulation (GDPR) (Sections 5.4.3 to 5.4.7). The GDPR Programme will define requirements and work with NGBI to define a design that is compliant with GDPR which will include role based access and other security model considerations.
- Data Management and Governance (section 5.4.1 and 5.4.2) and Azure Data Catalog (section 4.10). The approach for data management tooling enterprise wide and how the data management capabilities in Azure integrate with the enterprise approach will be considered subsequently.

This is also depicted in the diagram below. The NGBI Architecture and the technologies involved in it, are the scope of this document. The source systems and process involved of how they specifically are integrated into the architecture are out of scope.



1.1.2 Out of Scope

This document does not provide a design for the individual data marts/applications that will be built using the NGBI platform.

A number of current Business Intelligence platforms will be migrated and any inter-system connections re-pointed to the new platform where appropriate and achievable. The overall objectives will be the complete removal and decommission of those applications with the key targets being:

- EI
- Business Objects
- Data Services

The decommissioning of these applications and migration of existing platforms is out of scope of this document. The prioritisation and the methodology for migration and decommissioning will be set out in subsequent documentation.

A number of current downstream systems may be impacted by the migration, re-engineering or decommissioning of the above Business Intelligence platforms and these impacts are also out of scope of this document.

The following items below are also being considered out of scope and will not be included in this document, however, will be considered in subsequent design documentation in either the NGBI or other programmes.

- Azure Service Model (including DevOps), part of the Capgemini service transition as a result of the new run service contract signed at the end of 2017.
- Source System data integration patterns, including ingestion of SAP data

1.2 Intended Audience

This document is aimed at an I.T. technical audience that has some understanding of:

- Data Warehousing Principles;
- SCRUM Agile Principles;

- The Microsoft Azure Platform.

1.3 Overview

AWS would like to bring together data from on-premise data sources, external cloud services (e.g. weather and telematics) and applications data in to a single interconnected data model. The model will provide a central trustable source of information for the Business Intelligence solution, data exploration and applications integration. The objective is to build a central and secure source of any information formats that can scale storage and computing power to handle big data and provide business insights at real time for end-users through data marts, shared and self-service reporting in a cost-effective way.

The solution will be based on an Azure data services platform, the outputs of which will be:

- A data warehouse that will host the data used by the BI systems and data marts for the integrated applications
- A data model that brings together information from multiple sources
- A platform for data exploration
- A secure reporting solution capable of providing insights on multiple platforms and real-time reporting.

1.4 Proof of Concept

In preparation for the High-Level Design, a proof of concept has been conducted over the last two months. This PoC has utilised the technologies outlined in the document below, across three different business scenarios:

- BI Architecture: Telematics was chosen as a Proof of Concept candidate as it is an existing end-to-end BI solution, and, including it in the PoC allowed the team to prove capabilities of the platform in utilising different data types, and, ETL functionality
- Application Integration Architecture: Pollutions was chosen as a use case for application integration with the proposed BI architecture. This part of the PoC enabled the PoC team to outline design patterns to be used with Operational Data Store's in the future operating model.
- Data Science Architecture: Data Science was chosen as a PoC candidate to allow the PoC team to facilitate Data Science activities on the proposed BI architecture, and, demonstrate how predictive models can be productionised.

1.5 Key Drivers of the Solution

To architect a solution that:

- Brings together information assets (structured and non-structured) across the business within a single data lake store.
- Provides scalability in terms of computing power to aid the development and training of data models for data science
- Brings together both internal and external data sources
- Is capable of interfacing to other applications/middleware to share data models
- Allows the analysis of all types of information, for example – voice, video, text, content meta-data
- Quickly brings up virtual machines for data science and stand them back down on an ad-hoc basis
- Is flexible regarding changes in source systems in a way that doesn't affect the entire architecture
- Is automated as much as possible with deployment from development to production including code promotion, peer reviews and testing.
- Improves the performance for the current insight reporting solution

- Create a solution that allows self-service reporting
- Is built on a reporting layer that is compatible with multiple devices – tables, mobile, PC and thin client

1.6 Proposed Solution

The solution will use Azure Data Lake Store (ADLS) as a central repository for the data required by the Business Intelligence platform, applications integration data and to host the models created by Data Science tools. Data from multiple disparate sources – structured, semi-structured, unstructured and real time streaming data will be brought in to the ADLS. The live streaming data from events and telematics will be captured using Azure Event Hub, which is capable of capturing millions of events per second.

The data within the lake will then be cleaned and transformed leveraging the out of the box scalability and performance features of Azure Data Lake Analytics (ADLA). Azure Data Factory (ADF) will be used to ingest and prepare the data for consumption by the data science team, self-service users and reporting tools. This process enables sustainable information quality, fast and cost-effective data processing and security throughout the enterprise.

Following the extraction and transformation of data within ADLS a data warehouse will be built and hosted in Azure SQL Data Warehouse. The Azure SQL Data Warehouse platform can easily scale the computing power and storage required excessively based on demand. Data will then be modelled using Azure Analysis Services from where it will be accessible by front end tools like Power BI and Excel for reporting. The reports developed using Power BI can be shared securely within the organization and are compatible with multiple platforms.

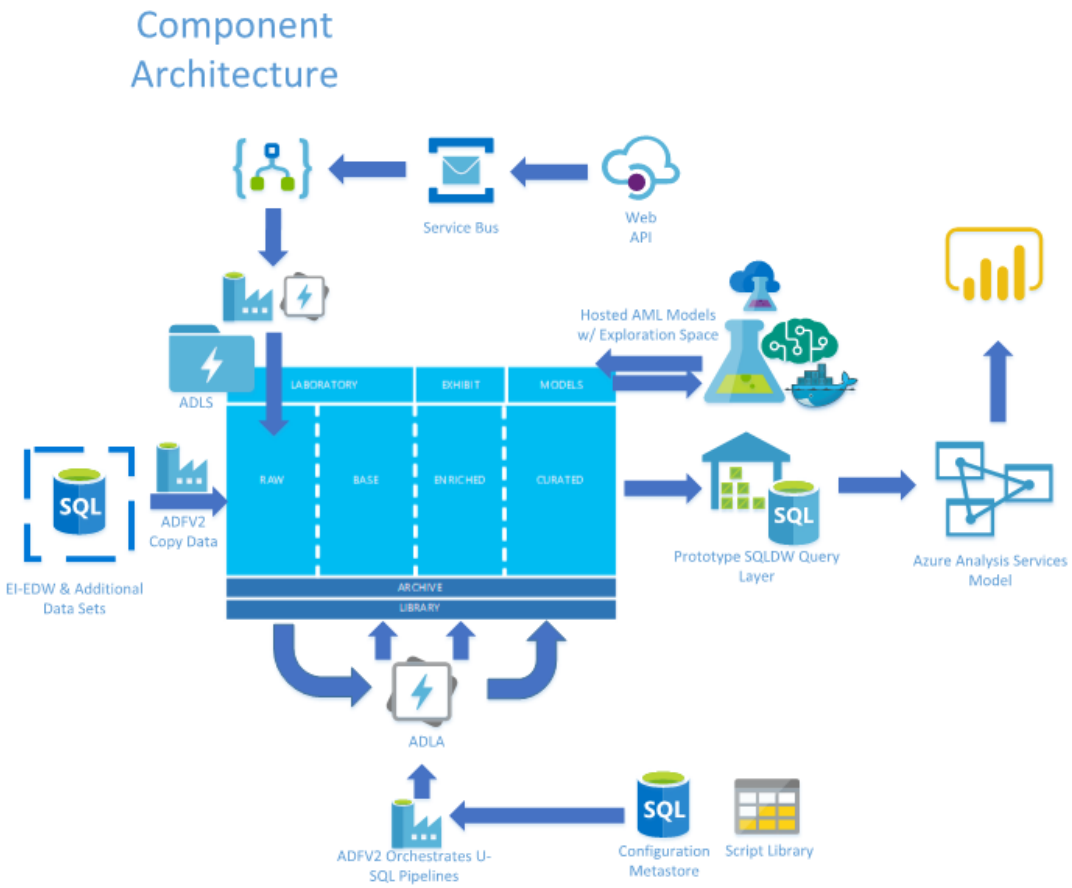
All components of the BI and Applications architecture will be built using Platform as a Service (PaaS) services in the Microsoft Azure cloud environment.

Some of the benefits of using PaaS solutions are:

- **Reduced costs** – No infrastructure to purchase and manage, pay only for what you use
- **Improved speed and agility** – Enable faster time to market by allowing development teams to focus on delivering business value rather than procuring, configuring any required infrastructure
- **Scalability** – The ability to easily scale the platform either in compute or storage as needed
- **Costs** - Reduced up front (capital) investment on hardware. Costs can be significantly reduced when less resource is required.

The key components of the solution include:

- **Azure Data Lake Store** – Central storage repository for all the data used by the BI platform and for data exploration
- **Azure Data Lake Analytics** – An on-demand analytics job service to develop and run massively parallel data transformation and processing programs in U-SQL, R, Python, and .NET over very large volumes data
- **Azure Data Factory v2** – Processes and combines the data produced by the on-premises and cloud data sources into trusted information.
- **Azure SQL Data Warehouse** – cloud based SQL data warehouse that seamlessly integrates with the ADLS providing a hub for the Pollutions data mart and tables required by the BI solution.
- **Azure Analysis Services** – provides a secure data model that supports a fast and straightforward way to build reports.
- **Reporting** – a reporting layer that utilises the data visualisation and collaborative features of Power BI.



1.7 Requirements Traceability Matrix

The functional and non-functional requirements of the NGBI platform are detailed in the accompanying Traceability Matrix document along with the response to how the proposed design will meet each requirement.

1.8 Architecture Implementation

Each component of the NGBI architecture will be introduced as and when it is required by each application that is being migrated and will not be stood up until it is needed. For instance, the Event Hubs and Streaming Analytics services to process streaming data will not be commissioned until there is a clear requirement to do so. It is envisaged that any required services for an application that have not yet been provisioned can be brought online by the team developing that application in the new architecture. This approach will give the teams a better understanding of the scaling required for each service.

Existing, or new applications and BI services should be developed end to end in the new architecture to deliver business benefit sooner, rather than an approach that would load all the data into the Data Lake ingestion areas and not give any user access to that data.

1.9 Solution Running Costs

The following sections detail the estimated running costs for the BI Architecture in Azure and are based on a number of assumptions. The complete list of assumptions and detail can be found in the document 'Anglian NGBI Technology Cost Model.xlsx' but the main points are:

- Pricing is based on Microsoft's recommended retail price but it is likely that Anglian will pay a reduced amount;
- The pricing costs for each component are correct as of the date of publication of this document and are likely to change in the future;
- The following prices are based on a scaling and sizing factor that assumes that all EI migration has been completed;
- The costs are based on MS Azure services only and do not include any other products such as Attunity Replicate.

1.9.1 Production Environment

The following table shows the estimated running costs for each component in a production environment:

Architecture Component	Daily Cost (£'s)	Monthly Cost (£'s)	Yearly Cost (£'s)
Data Lake Storage	27.78	833.28	9,999.36
Data Lake Ingestion	16.81	504.27	6,051.25
Data Lake Curation	32.81	984.20	11,810.35
Data Lake Preparation/Analysis	20.45	613.44	7,361.28
Data Catalog / Discovery	25.00	750.00	4,500.00
API Integration	16.80	504.00	6,048.00
Data Science Development	6.24	187.20	2,246.40
Streaming Data	7.75	232.47	2,789.64
Data Warehouse	37.61	1,128.24	13,538.88
Semantic Models	80.80	2,424.14	29,089.65
Presentation	250.00	7,500.00	90,000.00
Metadata and Audit	0.12	3.60	43.20
Total	**522.17**	**15,664.84**	**183,434.81**

1.9.2 Non-Production Environments

For non-production environments such as Dev/Test and Pre-Production a ratio has been applied to the production costs to determine the costs for these environments. The reasoning behind this is that non-production environments will not be required to have all the storage and compute time as production. For instance, it is not envisaged that Dev/Test will have automated analytics jobs running continuously, just on-demand as the developer/tester requires them. Similarly for pre-production it is not advisable to have all the data up to date and analytics jobs running to the same frequency as production. It is anticipated that pre-production data will be refreshed from production and compute jobs started just prior to a release to production depending on release cycles. The cost of user licenses for Data Catalog and Power BI have also been removed from the non-production estimates as they are already covered in the production costs.

The following ratios have been applied for non-production environments:

- Dev/Test – 20%
- Pre-Production – 50%

The following table shows the estimated running costs for non-production environments:

Environment	Daily Cost (£'s)	Monthly Cost (£'s)	Yearly Cost (£'s)
Dev/Test	49.43	1,482.90	17,794.80
Pre-Production	123.58	3,707.40	44,488.80
Total	**173.01**	**5190.30**	**62,283.6**

1.9.3 Total Costs

The following table shows the combined total estimated running costs for the NGBI architecture:

	Daily Cost (£'s)	Monthly Cost (£'s)	Yearly Cost (£'s)
NGBI Architecture	695.17	20,855.10	250,261.20

These costs have been estimated based on the amount of storage and compute required to process all data once the full EI migration has occurred. It is anticipated that year 1 costs will be significantly lower as not all data assets and functionality will have been migrated.

1.10 Risks, Assumptions & Dependencies

1.10.1 Risks

Below are the risks and proposed mitigation strategy, for each of the risks that have been identified around the architectures described in this document.

- **Risk:** The current Azure BI stack is an extremely fast-moving environment with many of the technologies on monthly release cycles of new features. This may result in the "best practice" approach changing regularly.
- **Mitigation:** The decoupled, "plug and play" style of this architecture helps ease the pain as new/different technologies can be added/removed and replaced if needed. Also, the Agile approach to delivery helps promote the change in this way to allow process to be change on smaller cycles rather than the long planning phases seen in a Waterfall approach.

- **Risk:** Data Factory V2 is currently in preview.
- **Mitigation:** An exact date of when Data Factory V2 will be Generally Available is not yet available but it is to be expected in the next few months.

- **Risk:** The costs of Platform as a Service technologies is inside of Microsoft control
- **Mitigation:** Historically we tend to see the costs of PaaS technologies come down over time, but in theory they could just as easily go up.

- **Risk:** True High Availability or Disaster Recovery is not available for some of the cloud technologies.

- **Mitigation:** Details at the individual technology have been specified on how to manage this in the Solution Components Management section of this document.

- **Risk:** The technologies in the architectures described in this document are largely new to AWS and therefore significant training will be required by the company on the new process and ways of working.
- **Mitigation:** Appropriate training should be provided.

- **Risk:** Data Lake can become messy/out of control if rigorous folder management is not adopted.
- **Mitigation:** Folder Structure framework will be implemented from the start.

- **Risk:** If Data Catalogue isn't used appropriately by everyone then it can quickly become out of date and in turn become useless.
- **Mitigation:** Process and training should be implemented from the start on how Data Catalogues purpose and how to use it.

- **Risk:** A well-defined support model is not currently in place for all the technologies used in this architecture.
- **Mitigation:** A model needs to be defined with support suppliers and AWS.

- **Risk:** If data is downloaded to a non-corporate device, or, used incorrectly by a malicious user, there is a risk that the organisation may fail to meet GDPR compliance.
- **Mitigation1:** GDPR requirements are to be understood when available – likely to be in the DLD phase of the ngBI Foundations Project
- **Mitigation2:** Anglian Water will define a policy to govern downloading of data to non-corporate devices, and, this policy will be adhered to by the ngBI Programme

- **Risk:** AW is currently transitioning to a new SIEM service provider. There is a risk that the new SIEM may add complexity to the solution.
- **Mitigation:** Engage with Airbus to understand the requirements of the new SIEM solution as soon as these requirements are available, and, incorporate the interaction with the SIEM into the detailed design of the platform

1.10.2 Assumptions

Below are the assumptions associated with the architectures described in this document.

- Data from SAP will be placed into the Data Lake via Attunity Replicate.
- Attunity Replicate will provide a datetime stamp of when rows were added to the Data Lake.
- Azure Data Lake Store, curation and governance approach will be defined during the build phase of the project.
- Sufficient bandwidth will be provided for moving data from on premise applications to the cloud.
- All required data sources will be compatible with Azure Data Factory or accessible via a WebAPI.
- Power BI is sufficient for all self service and enterprise reporting requirements.
- B2B access of data or reports requires that outside users must have an Azure Active Directory account in their own tenancy or to be added to the AWS Azure Active Directory.
- As performance requirements are as yet undefined, it has been assumed that the architecture can be scaled to meet the requirements within a reasonable cost parameter.
- The new AIRBUS SIEM solution will be able to support Microsoft Azure Cloud-based solutions

1.10.3 Dependencies

- In order to correctly define the solution in the Detailed Design phase, the ngBI Programme is dependent on AW to define policy to govern the scenarios where users of the ngBI Platform may choose to download corporate data onto unmanaged (3rd party or personal) devices. Further, if this practice is to be allowed, a retention policy will also be required.

Assumptions

2 Solution Design

2.1 Solution Requirements

This document will only cover high level requirements for each architecture.

Detailed requirements can be found in the Traceability Matrix, along with comments around how the architectures detailed in this document would support the requirements.

2.1.1 Modern Business Intelligence Architecture

Below are the high-level requirements around the BI Architecture, that the proposed solution will aim to cover:

- A centrally accessible data platform for everyone to gain the insight they require.
- A cloud based platform that is future proof, with a plug and play approach to support all types of data.
- The architecture must be performant and scalable.
- A platform that provides the ability to support a modern DevOps strategy to turn requirements into releasable content faster.
- A platform that will assist the company in becoming GDPR compliant.
- The reporting solution must support a self-service operating model.
- The solution must be able to ingest data from all current data sources that form the EI Warehouse.
- The solution must be able to cleanse and manipulate data to (at least) the same standard as that carried out in the current solution.
- The solution must be able to provide the data to an enterprise reporting solution, in a timely, usable fashion.
- Data must be accessible from systems (applications) in the business.

2.1.2 Application Integration Architecture

Below are the high-level requirements around Applications Integration Architecture, that the proposed solution will aim to cover:

- The architecture must support near-time reporting of the applications data.
- The architecture must support reporting of history from the applications data.
- The architecture must be able to report on data sets made up of the applications data combined with data from other source systems.

2.1.3 Data Science & Exploration Architecture

Below are the high-level requirements around Data Science, that the proposed solution will aim to cover:

- A platform that provides the space and capability to explore source data and curated data sets.
- A platform that can be easily scaled to give appropriate power for work on large data sets.
- A platform to support spinning up and throwing away new environments for work on ad-hoc data sets.
- Support for a growing list of tools and flexibility in languages used.

2.2 Design Principals & Objectives

The following are the standard set of design principals used by Adatis. We have evaluated the solution against these principals and recorded any deviations. This displays current state and will be updated throughout the lifecycle of the POC and Design phase.

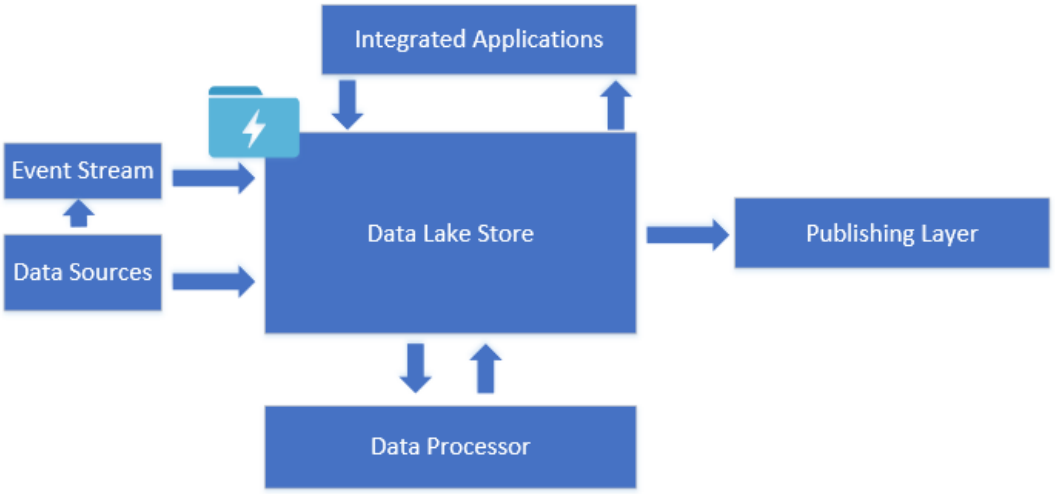
Area	Principle	Description	RAG	Comments
Architecture	**Automation over Hand-Cranking**	Automate as many processes as possible and minimize the role of customization and manual effort	GREEN	Automation tools have been identified and will be proved as part of the POC
Architecture	**Big Picture Planning**	Information Technology decisions are made to provide maximum benefit to the enterprise as a whole.	GREEN	Architecture has been reviewed for future expansion & integrations
Architecture	**Business-Driven Development**	The technology architecture is based on a design of services which mirror real-world business activities comprising the enterprise (or inter-enterprise) business processes.	GREEN	Architecture is driven by actual use-cases to underpin any requirements.
Architecture	**Failure's Aren't Fatal**	Enterprise operations are maintained despite system interruptions	AMBER	Logging & Failure paths are supported by the POC architecture but the requirements for the management system will be fleshed out during the pilot phases
Architecture	**Loose Coupling**	A modular system is one that is structured into identifiable abstractions called components. Each feature of the architecture should be built to perform its purpose and isolated from other components.	GREEN	Solution consists of separate components which can be individually scaled/switched out as the solution evolves.
Architecture	**Polyglot Solutions**	Solutions should utilise the tools and technologies most suitable for each task. Solutions will likely use several languages in their implementation. However, technical diversity has a non-trivial maintenance overhead - the solution should endeavour to only use one tool for each type of work to minimise this	GREEN	Appropriate tools selected – potential future roadmap to be identified.
Architecture	**Relationships Built on Trust**	Information should be complete, accurate, free of duplicates and have relationship integrity. It must be easy to access regardless of where it is stored. Information should be easily available to those who should have access to it.	AMBER	Data quality routines implemented, further data cataloguing tools to be confirmed
Architecture	**Reuse > Build > Buy**	Solution components should be based on existing patterns & libraries where possible, however third-party reliance should be avoided	GREEN	Existing Adatis patterns utilised within lake design & data processing. No third-party reliance involved across solution.

Area	Principle	Description	RAG	Comments
Architecture	**Secure by Design**	We shall design to protect the value of the systems and information assets of the enterprise	AMBER	Standard best practise security patterns implemented – deeper security review of each user/application will be conducted during the preparation of the DLD.
Architecture	**The Right Tools for the Job**	The technology landscape is changing rapidly and sometimes the less mature technology is the more relevant to modern approaches. We take a pioneering approach, evaluating technology as it emerges to see if it is a more relevant fit for our current purposes - as long as it is a better fit than our current approach!	GREEN	Technology selections researched & justified – decisions documented in later sections.
Data & Governance	**All Data needs an Owner**	In the days of GDPR, we need to be sure that data is allowed to be in different places and know who is responsible & accountable for its usage. All systems should be designed with governance and data stewardship at its core.	GREEN	Anglian governance process supported by architecture
Data & Governance	**Always Question Kimball**	Kimball data models are great in many cases, but a lot of the design is to increase performance in a traditional RDBMS, with modern parallelism-based tools a de-normalised flat table might perform better. Kimball is still very relevant, but we should make sure we're not just using it out of habit	GREEN	Requirement for Analysis cube strongly suggests Kimball model. New data sets to be evaluated as they are introduced into the ecosystem.
Data & Governance	**Cat Skinning**	There are many ways to skin a cat, if we are performing similar processes, we should adopt a standard approach. Templating and pattern sharing should be embedded within the core of the build process	GREEN	Templating already in use within the POC architecture
Data & Governance	**Do It Yourself**	Modern analytics should be delivered in a timely manner, opportunities are often missed where requests are fed through a central team. All analytics systems should aspire to allow users the freedom to access data when and where they need it	GREEN	Variety of business users modelled in requirements, from interactive dashboards (Power BI) to Data Science exploration frameworks
Data & Governance	**Just the Right Amount of Control**	Governance is important - it helps control project costs, ensure the right solution is delivered, protect sensitive data and much more, however it can also be overbearing and stifle innovation & agility. Governance should be appropriate to the solution and agreed in advance.	AMBER	Agile processes will be implemented for development – process should be applied to the requirements backlog for build prioritisation, to improve the velocity of business value delivery. For the avoidance of doubt, prioritisation will be relate to the order of which data-mart will be chosen to utilise the new

Area	Principle	Description	RAG	Comments
				platform. Non-functional requirements including Security, will not be compromised by this prioritisation, unless by express, separate agreement with the relevant stakeholders
Data & Governance	**User-focussed Data Modelling**	Data Models produced should be clear and understandable and in a structure that makes sense to the business user rather than developers	GREEN	Power BI & Analysis Services encourage true semantic modelling to improve business user experience
Infrastructure	**Cattle not Pets**	The cloud approach to infrastructure acknowledges that physical hardware is fallible and short-lived. Any solutions should be deployable to new infrastructure with minimal disruption, with infrastructure configuration stored as part of the solution assets	AMBER	Solution relies on fault-tolerant infrastructure with minimal configuration. ARM Templates & CI/CD pipelines to be delivered in a later phase.
Infrastructure	**PaaS not IaaS**	The aim of any data analytics platform is to deliver business value - any time spent managing/configuring hardware is time not spent delivering value. Solutions should aim for a maintenance-free design where possible	GREEN	All automated components are PaaS and support has been evaluated. Only potential IaaS is the Data Science VMs, which will be designed to be replaceable by VM image.
Infrastructure	**Quiet Clouds**	Unused capacity is wasted money - any infrastructure design should scale elastically to meet capacity demands, rather than over-provisioning & providing redundancy	GREEN	Individually scalable tools selected for architecture, baseline pricing tiers will be recommended which can be reviewed over time
Infrastructure	**Say what you see**	Naming conventions are very important, but can sometimes make working with complicated infrastructure very complicated. Any naming conventions should take readability and comprehension into account - if you can tell what the component is and where it fits into the architecture, we're doing things right.	AMBER	Anglian naming convention adopted – component types are clear but the intention is not necessarily.
Operations & Management	**Can't see the Logs for the Forest**	Solutions should have a central logging & auditing system. Logging is great but if the logs aren't visible & available, they're worthless	AMBER	Central logging system proposed but requirements and full design are yet to be implemented.
Operations & Management	**Standardised Systems**	The Adatis framework is largely seen as a development accelerator, allowing us to deliver solutions faster than building it from scratch. However, it also allows for consultants who were not involved in the original build to have a good idea where to find things, how functionality has been implemented etc. If there's a framework feature for the functionality being implemented, it should be used.	GREEN	"BI System" metastore design implemented, along with standard Data Lake Framework & loading patterns

2.3 Design Overview

The diagram below depicts a conceptual component overview of the solution. The diagram shows 4 main technical areas of the solution surrounding the central hub of data (Data Lake Store).

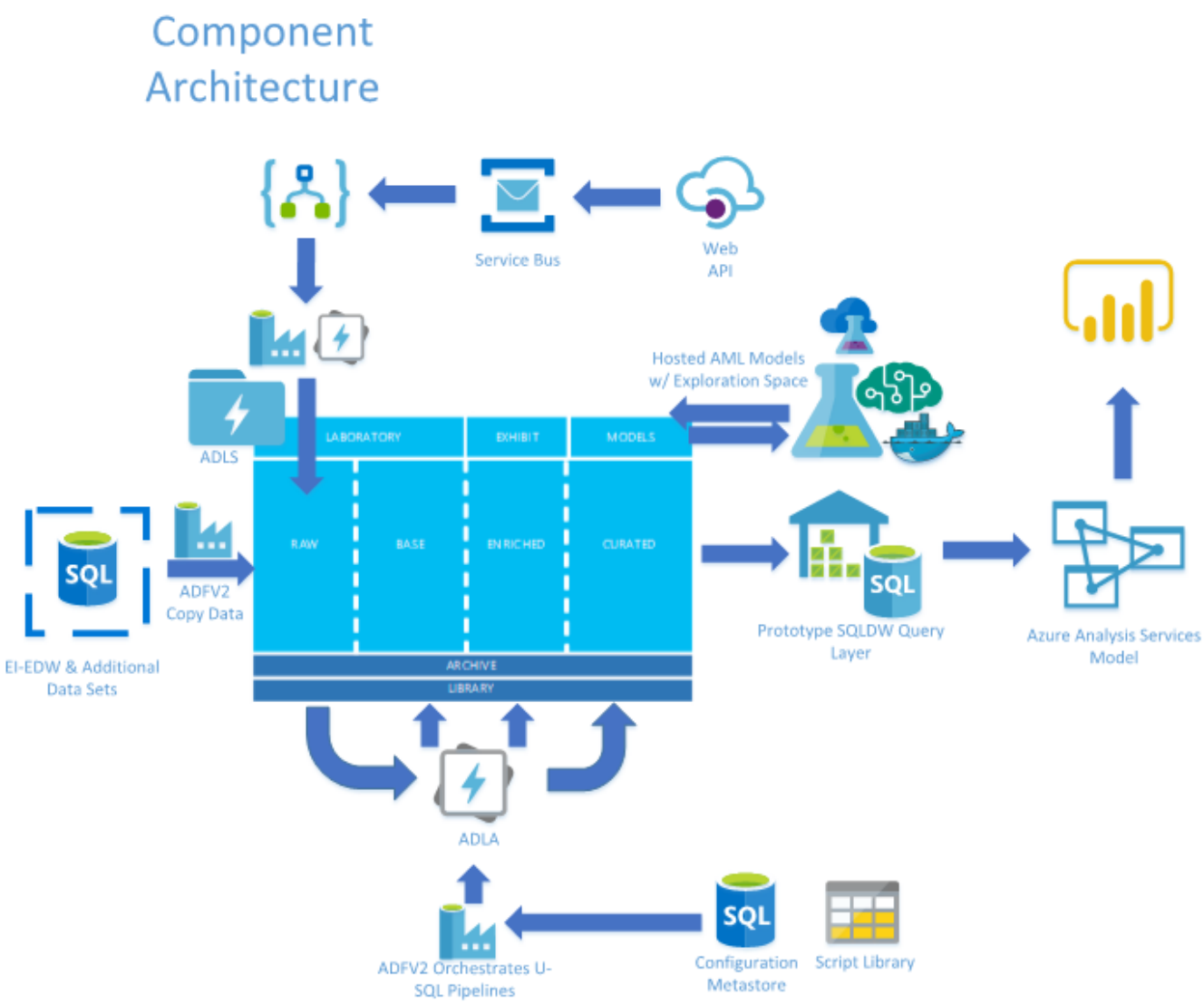


Below is a brief description of each of the technical areas:

- **Data Lake Store** – this is the central store of data, all analytics data moving through the AWS analytics systems will be channeled through here for storage, cleaning and presentation to downstream systems.
- **Data Sources** – various source systems (on-premise, API's etc.) will provide data to the Data Lake Store, landing it in a RAW area for further processing.
- **Data Processor** – this component provides our transformation layer, providing the compute & logic required to process data, perform quality and validation processing and to produce data sets for the other areas.
- **Integrated Applications** – event streaming is used to allow real time data import and processing, it can be used as a messaging endpoint for AWS Applications which will then control the feed of data back into the Data Lake Store. The POC will be demonstrating this functionality using Pollutions data only.
- **Publishing Layer** – this area will extract processed data out of the Data Lake and process it into a semantic layer, designed for fast analysis and performant aggregations. This semantic layer is then presented in interactive dashboards through reporting tools such as PowerBI.

2.4 Design Breakdown

The diagram below depicts a detailed component overview of the solution with the high-level technologies used in each area.



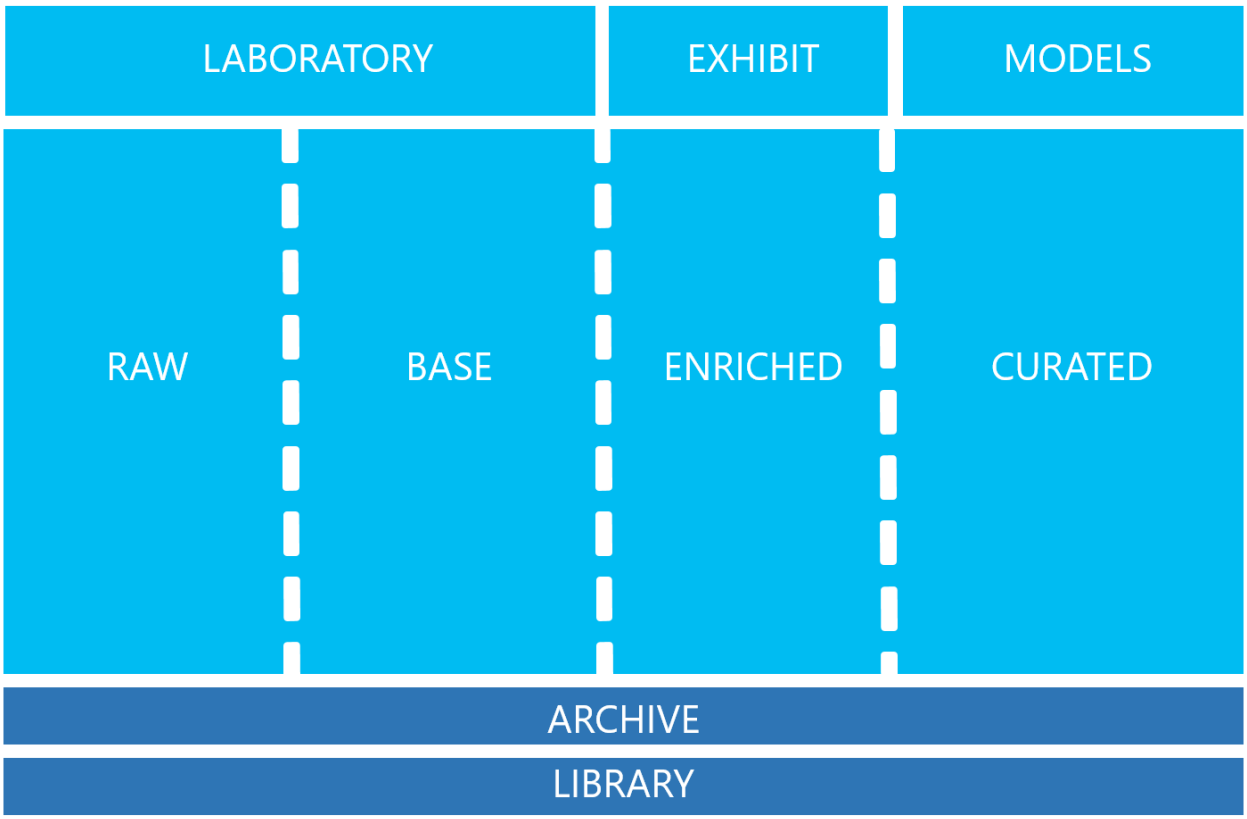
The overall architecture diagram contains many moving parts. For simplicity, we will discuss each technical area in further detail, discussing the component parts, the patterns implemented and any integrations with other technical areas.

For more detail on the individual technologies please see Section 3, and for information and the management of the technologies please see Section 4.

2.4.1 Data Lake Storage Layer

The central part of the architecture shows the structure that has been applied to the Data Lake Store component. The technology itself hides much of the complexity, allowing users to treat the storage layer as a network share – creating folders & files and applying read/write security privileges as necessary.

Adatis have recommended a base structure to these folders that underpins our data ingestion processes – these folders segregate data of different stages of preparation and support both the security layer and a logical separation of data areas. The Data Lake Framework is as follows:



The Data Lake is separated into 3 main areas for data storage: Data Science (top), ETL (middle) and Archive (bottom). These areas are described below along with how they break downs into sub areas.

- **Data Science** – This is a dedicated area for the Data Science team to draw in data from various other locations of the lake, create models and experiment with data.

- **Laboratory** - This area has fewer access controls than others and is intended as an experimentation area for Data Science teams. Each user/team/project (depending on preference) has its own folder, where temporary data sets can be created and models tested.
- **Exhibit** - Sometimes, Data Science teams produce one-off data sets than can add value to other areas of the business, the EXHIBIT is intended to share these data sets. EXHIBIT data sets are not created by automated processes, and do not have the same level of validation/control as those in the CURATED area and so are stored separately. These files need documentation around their intended purpose and the data contained within
- **Model** - Not everything within the Data Lake is a data set – Data Science team often produce models which can be serialised and stored within the Data Set for reference by other applications. Adatis have designed a model management structure and process which utilises this MODEL area.

- **ETL** – This is the central data repository for data being loaded into the Warehouse, as data is manipulated it moves through the 4 areas stated below.

- **RAW** - The Raw layer acts as an immutable delta, regardless of the source system, file extracts are landed into RAW in their original format, using folder/file-based vertical partitioning to allow for easy querying/access of history & change.

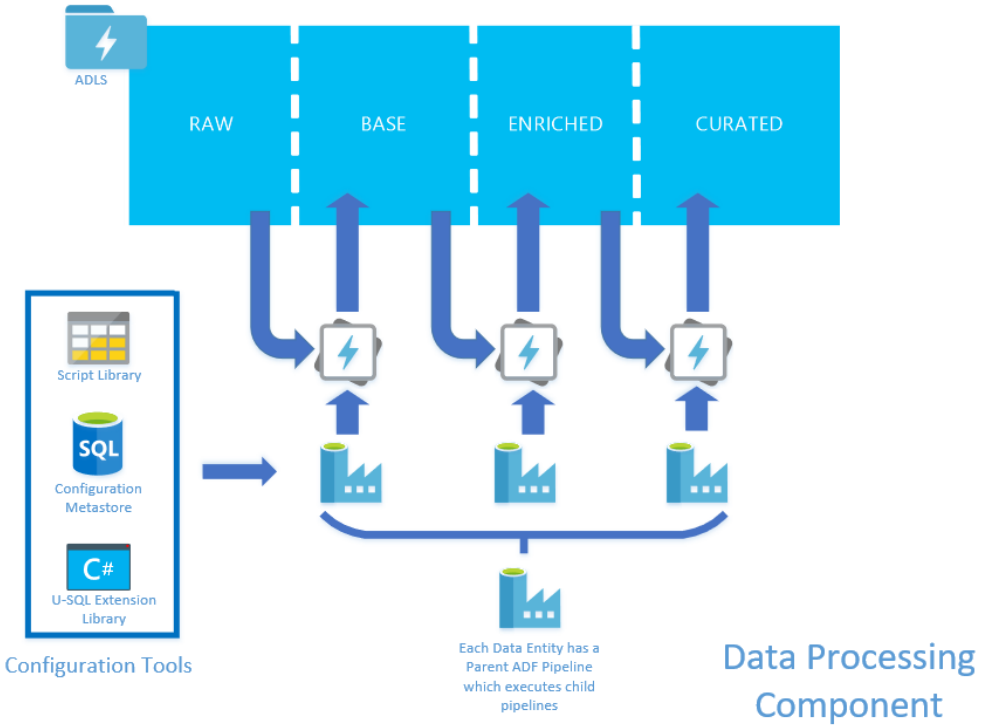
- **BASE** - Automated processes are used to pick up a file landing within the RAW area and create a "clean" copy within the BASE folder. Rules can be automatically applied to enforce data types/formats, remove special characters, apply encoding and more
- **ENRICHED** - The ENRICHED area serves as a staging area before loading data into the final data models. Data Transformation scripts apply business logic to combine files from BASE, apply business logic, add derived columns and any other transformations required.
- **CURATED** - This area contains data that is ready for consumption by business users and other downstream applications. The data contained within has been cleaned, validated, checked and prepared for usage. All data held within CURATED should have appropriate documentation within the chosen data cataloguing system – this should contain business definitions, data profiling and a nominated business expert.

- **Archive and Management** – This area contains the archived data and the management components of the Data Lake

- **Archive** - Sometimes, data held within the Lake is no longer of immediate business value. If we are sure the data is only going to be accessed in rare occasions, we can utilise Cold/Archive blob storage, which comes at a much-reduced cost at the price of higher per-access charges. This can act as an extended-section of the lake and is still accessible by our ETL & compute tools.
- **Library** - Many of our automated processes use template files, dynamically generated scripts, validation result files and more. The LIBRARY section holds these various files which are an intrinsic part of the data processing tools and utilities used by Adatis
- **Catalogue** – This is the deployment area for any scripts that are created in Azure Data Lake Analytics.
- **System** – This is the backend system folder created for Azure Data Lake.

2.4.2 Data Processor

The data processing area of the solution consists of pulling data from the RAW area of the lake and then orchestrating the movement, cleansing and creation of data sets through the other phases of the Data Lake Store. Each stage of data processing is an isolated, re-runnable component.



The technologies used in this area are listed below:

- **Data Lake Store** – The data lake is the repository for the data.
- **Data Lake Analytics** – Data Lake Analytics stores the logical structure of the data stored in the data lake. It is also used as the store to hold U-SQL stored procedures that are used to extract data from one part of the lake and move it to others, as well as perform business logic and any additional columns or meta data columns.
- **U-SQL C# Extensions Library** – U-SQL procedures can call C# libraries to assist in data manipulation. These are a physically stored within Data Lake Analytics.
- **Data Factory v2** – Data Factory is the orchestration tool used to execute the U-SQL stored procedures. It will also call the logging procedures to log details about the load progress.
- **Azure SQL Database** – The SQL database stores the logs on the data processing performance.

2.4.3 Publishing Layer

The publishing layer extracts complete data sets (as flat files) from the data lake using the PolyBase features in Azure SQL DW and presents them as tables to Azure Analysis Services (AAS). A Tabular model hosted in AAS provides an easy to use business focused platform for end users to connect to data. Power BI or Excel can be used to connect to AAS to view data and create reports.

The technologies used in this area are listed:

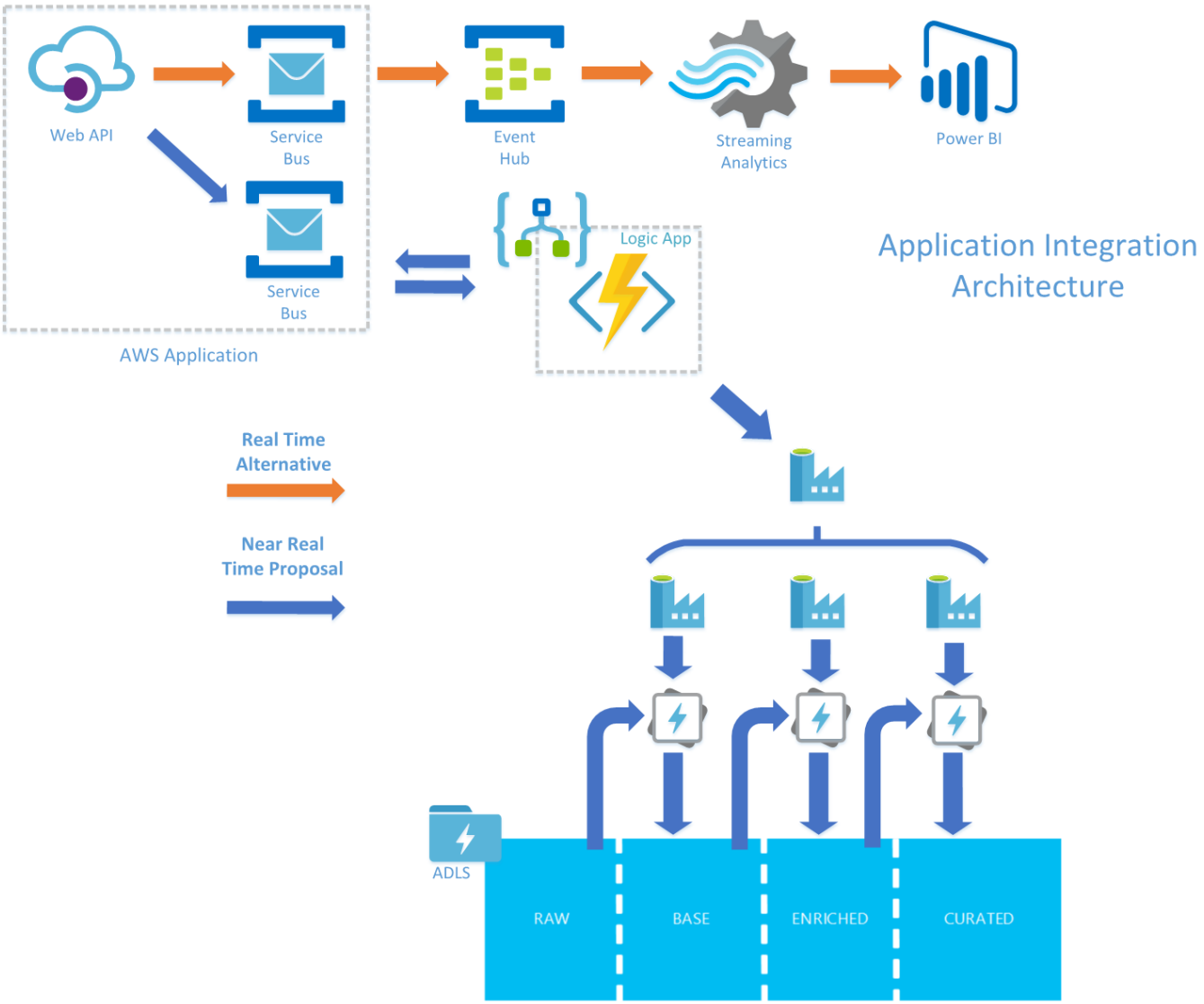
- **Data Lake Store** – The data lake is the repository for the data.
- **Azure SQLDW** – SQLDW in this solution is primarily used as an abstraction layer, an interface between the already complete dataset flat files in the Data Lake and providing this information as a table to Azure Analysis Services.
- **Azure Analysis Services** – AAS is the semantic layer in the solution, the transition from the technical workings of data into the business areas and structure known and understood by end users.
- **Power BI** – Power BI is the industry flagship reporting tool for self-service, real time and interactive analytics.

2.4.4 Application Integration

Anglian Water currently has many custom-built web applications across the business. These applications are hosted independently and operations fall outside of this HLD. The Application Integration Architecture describes the way in which data from the applications will be ingested into the Data Lake and reported on in near real-time.

The technologies used in this area are list below with descriptions on the purpose and use of each.

- **Azure Service Bus** – Messages from the WebAPI's (currently built into the applications) will post messages to the Azure Service Bus. This is already common in the applications.
- **Logic Apps** – A Logic App will then be used to poll the Service Bus, and when a message appears kick off the Data Factory.
- **Azure Data Factory** – The Data Factory will take the messages from the Service Bus and load the data through the Data Lake Store applying any cleansing and business logic that is required.
- **Data Lake Store** – The data lake is the repository for the data.
- **Azure SQL Data Warehouse** – Once in its final state, SQLDW will use PolyBase to surface the data as a table.
- **Azure Analysis Services** – AAS will use the tables in SQLDW to extract the data into an Analysis Services Tabular model.
- **Azure Automation** – Azure Automation will be used to process the Analysis Services model.
- **Power BI** – Power BI will connect to the Analysis Services model to display data to end users.



The Application Architecture provides an independent approach to managing, scaling and loading individual applications. Each application will come with its own set of components allowing them to be controlled uniquely for the requirements of the application.

The standard architecture can provide near real time reporting as it loads data and combines to data with other sources, on a fast-paced schedule. If real time reporting is required then with a small change to the architecture Event Hubs and Stream Analytics can be used to provide this functionality, this design can be seen on the diagram as the Real Time Approach.

When using Streaming Data Sets in Power BI, it can prove difficult to combine data sets together.

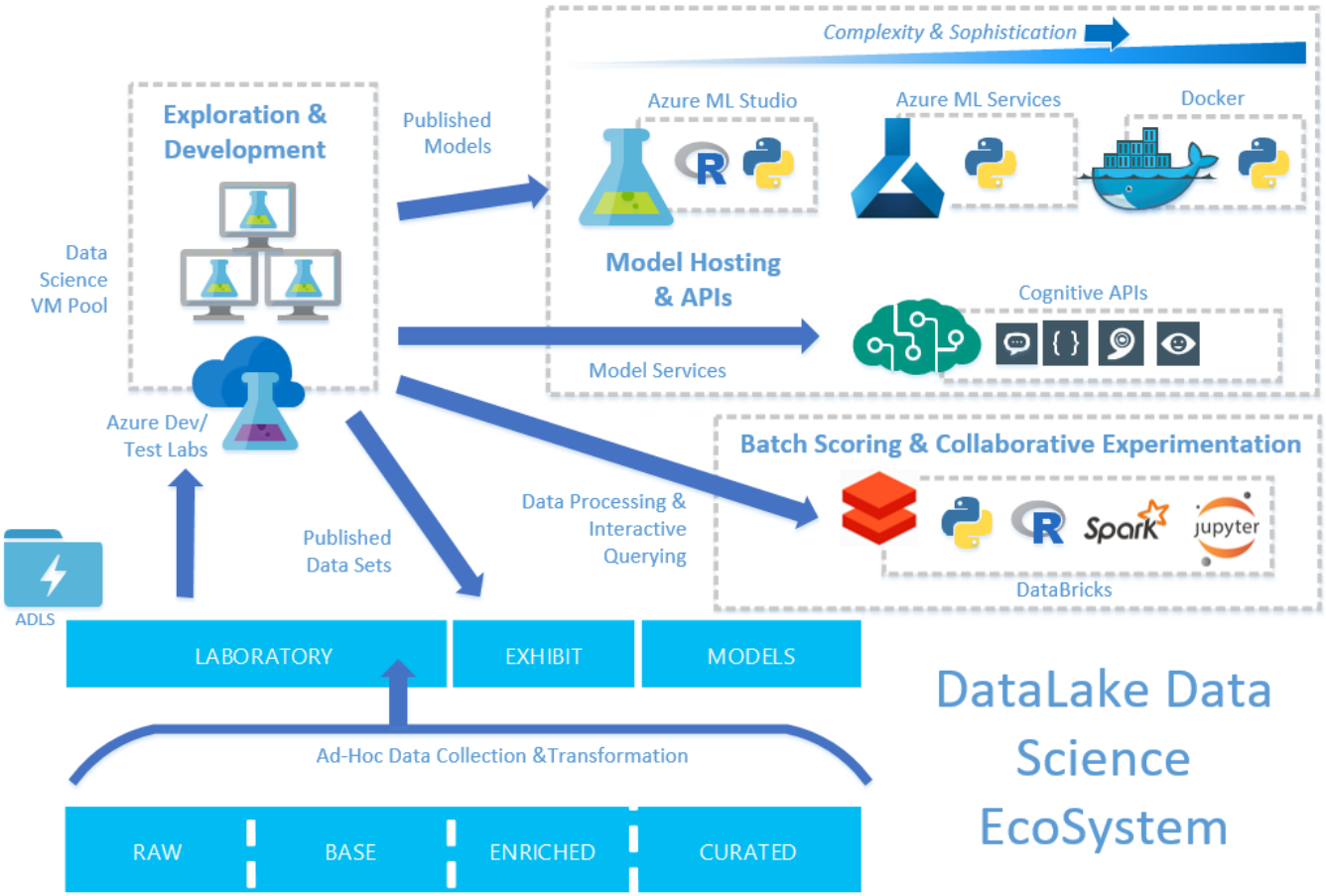
Utilising the Azure Service Bus and having the Web API drop messages here allows us to create an architecture that requires little change to the current applications and their process. The Web API's currently write to an Azure Service Bus to allow other systems receive data from the applications.

The Azure Service Bus also decouples the applications from the reporting side, therefore if failure occurs on either side it will not create knock-on effects further downstream.

This Applications Architecture also provides benefit in how it falls in line with data storage and by utilising the Data Lake structure it allows for data exploration of the source data via Data Scientists without effecting the primary purpose of real time reporting.

Storing data in Azure Analysis Services gives end users the freedom to explore the procured data sets in Excel and Power BI to generate their own reports, to help gain the insights required outside of the predefined reports.

2.4.5 Data Science



The technologies used in this area are list below with descriptions on the purpose and use of each.

- **Data Lake Store** – The data lake is the repository for the data.
- **Data Science Virtual Machines** – These are used by the data scientists for the client-side development of models
- **Azure Machine Learning Studio** – This is used for the development and hosting of lightweight, simple models
- **Azure Machine Learning Workbench** – This is the client-side project management environment, providing source control integration and model performance benchmarking
- **Azure Machine Learning Experimentation Services –** An environment management service used to provision remote docker containers for isolated model runs
- **Azure Machine Learning Model Management Services –** A managed docker cluster with API wrapping service – this is the default hosting location for machine learning models
- **Docker –** Configurable model hosting environment, provides much more compatibility than AML Model Management but requires much more code to deploy and manage

- **Azure DataBricks –** PaaS Spark environment, used for the running of memory-intensive batch scoring models
- **Cognitive APIs –** Pre-built models providing generic services such as face recognition, sentiment analysis and language detection.

The Data Science environment covers two main areas, experimentation and model management/hosting. The two solutions are best discussed and described separately.

2.4.5.1 Data Science Experimentation Platform

The aim here is to give the data science team the freedom to experiment, innovate and collaborate to encourage the best models. Many of the traditional development practices are relaxed, with the data scientists able to try out a variety of approaches without restriction.

The ***Data Science VMs*** are pre-packaged with a variety of common tools and libraries, essentially a ready-made development platform for a data science developer. Additional libraries and frameworks can be pulled down onto the machine, allowing the scientists to keep up with the evolution of statistical languages – it is far better that they can access open source libraries to achieve their aims than having to build everything from scratch.

The ***Data Lake Laboratory*** area acts as a sandbox for experimentation – the data science team can structure this storage as they wish and should be free to build temporary data sets within this area. These datasets are outside of the regular governance process as they are not visible to business users, instead the data science team governs its own storage consumption and is responsible for archiving/deleting old datasets as projects close. The team can build these datasets using Data Lake Analytics, as well other areas of the Lake Ingestion process, or they can build their own data preparation scripts in python, Scala or other languages, via the ***Data Science VMs***.

Finally, the ***Azure Machine Learning Experimentation Service*** is used to develop more complex models and run them in different environments. Initial scripts can be written in python, with a full set of dependencies, libraries etc. These models can then be run locally on the machine, in a local docker container, or automatically wrapped and ran on a remote docker service. This allows for an evolving testing process, getting models ready for production deployment and ensuring the model is in the right format.

2.4.5.2 Data Science Model Hosting Platform

Lightweight models - Once a model is ready for hosting, we have three main options for where it can be hosted. Our first is the light-weight solution of ***Azure Machine Learning Studio***. This is a very user-friendly hosting service where models can be deployed and pushed to a production state. The service automatically wraps the model with a basic API, however there are many limitations about what can be done within the service and is often seen as too limited for a serious Data Science team.

Default Model Hosting - Our next method would be the default model hosting solution. **Azure Machine Learning Model Management Services** is a new Azure Service designed to simplify the model hosting lifecycle. The management service is essentially a managed docker container service, with several telemetry & API tools surrounding the deployed images. A model can be deployed directly from within Machine Learning Workbench, at which point a new docker image is created and deployed to the service. The service monitors incoming scoring requests and maintains performance metrics which can be pulled into central logging systems.

Models deployed to the Model Management Service have to confirm to a default structure, with several pre-defined endpoints exposed to the service. There is also a limited language set that is currently supported by the service, although further languages are planned for later release.

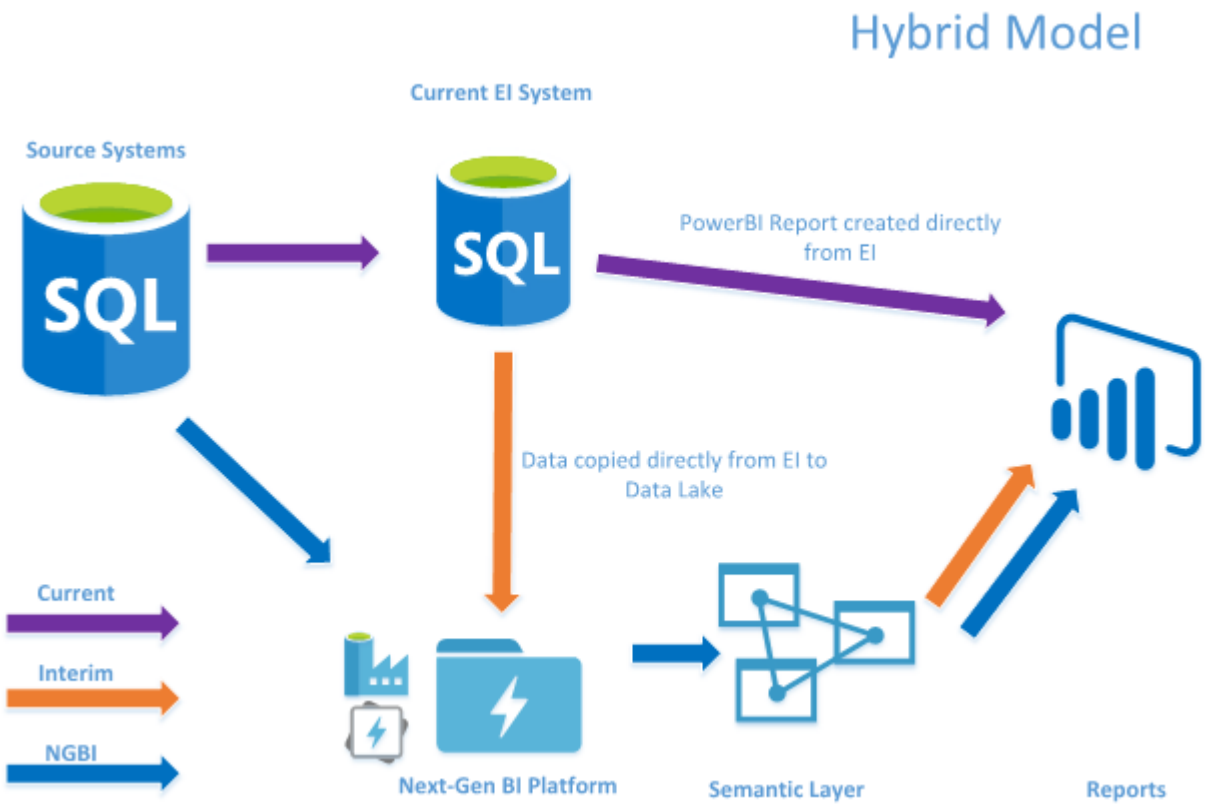
Complex Model Hosting – there will be occasions where the language limitations, or pre-defined structure requirements of Azure Model Management Services is too limited for a very specific challenge. In these cases, the model can be deployed to its own, bespoke docker container. This can be generated from the images defined by the Experimentation Service, however it would not be deployed under the managed container service provided by the Model Management Service. These docker images would need to be managed for performance & scalability and any automated model re-training would have to be built manually.

Batch Model Hosting – The Data Science team do not have batch scoring models on their immediate roadmap. The Azure DataBricks proposition is in a very early preview state currently and wouldn't be a recommended part of a production environment. Azure DataBricks is expected to be generally available before the data science team is ready to productionise Batch scoring models at which point it would be the best hosting location for any batch scoring models produced by the data science team.

2.5 Hybrid Interim Architecture

With the aim on minimising the amount of technical debt created whilst the NGBI architecture is built a hybrid interim architecture has been designed to allow as end users to quickly be able to start using the NGBI reporting tools and fulfil new data requirements without creating the need for it to be all rebuilt when the NGBI solution is complete.

Below is a diagram showing the Hybrid Interim Architecture:



As detailed above, the proposed solution involves moving complete data sets out of the current EI Warehouse and into the RAW section of the Data Lake. The data sets will then be picked up and moved through the Data Lake and end in the Curated section of the Lake.

Analysis Services models and reports, can then be built on top of the curated data sets. What this provides is a stable ground to build these reports and models, as the Curated data sets will not change format after they have been fully implemented into the solution.

As data sets are rebuilt in the new architecture source tables can then be added to the RAW section of the Data Lake and then manipulated to create the data sets that were previously being extracted from the EI Warehouse. These data sets will then supplement what is currently in place in the Curated section of the Data Lake and will result in no further changes needing to be made to the Analysis Services model or reports.

This solution will not allow exploration of source data through the data lake.

As data becomes available in the Data Lake migration from the interim solution should be straight forward and require minimal changes to the model under the following conditions:

- Data structure does not change
- Data granularity does not change
- Data types do not change

Significant changes to the above conditions may result in further rework to be carried out.

It is not recommended that temporary data sets are created using Power BI datasets as migration to the new architecture will need to be carried out multiple times.

It is recommended that all interim solutions are sourced from Analysis Services models as the migration can be controlled from a single source by the BICC team.

3 Solution Components Technical Overview

This section contains a technical overview of each of the technologies used within the solution.

Each overview is broken down into the following sections:

- **Component Summary** – this will detail what the technology is and what it was built for, not in reference to this solution.
- **Use in architecture (BI, Applications, Data Science)** – this will describe how we are using the technology in each of the 3 business areas.
- **Alternatives and Reasoning** - This will describe why this technology was chosen and the alternatives that could be used.

3.1 Azure Data Lake Store (ADLS)

3.1.1 Component Summary

Azure Data Lake Store (ADLS) is an enterprise-wide hyper-scale repository for big data analytic workloads. ADLS enables you to capture data of any size, type, and ingestion speed in one single place for operational and exploratory analytics. It is essentially a wrapper around a Hadoop Distributed File System (HDFS), which is the traditional storage layer in Big Data processing architectures.

ADLS can be accessed using the WebHDFS-compatible REST APIs. It is specifically designed to enable analytics on the stored data and is tuned for performance for data analytics scenarios. Out of the box, it includes all the enterprise-grade capabilities—security, manageability, scalability, reliability, and availability—essential for real-world enterprise use cases.

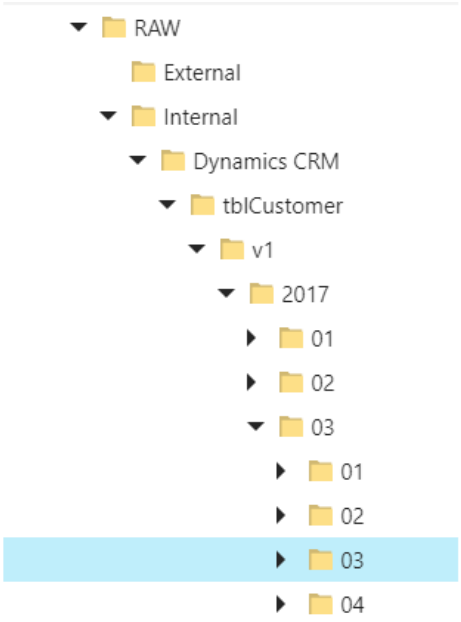
Using a distributed storage mechanism (WHDFS/HDFS) for data storage allows us to run MapReduce Style jobs which give massive performance gains due to parallelisation.

The performance benefits of using HDFS-based technologies are discussed under operational considerations, see section 4.

3.1.2 Suggested folder structure for: RAW/BASE/ENRICHED

As these three areas of the lake contain delta files, meaning they do not store a complete data set in a single file we need to be able to track and structure how we store these files to be able to identify the order (day) of which the files arrived.

Below is a diagram depicting the recommended structure of these areas.



A description of each level is described below with examples of each level.

- **Lake Area –** e.g. RAW, BASE, ENRICHED. This is the top of the structure, and all sections below are repeated across each of the sections in this level.
- **Data Source Type** – e.g. Internal, External, Open Source. This is a categorisation of the data source to help navigate through the lake when looking for a particular data set, also gives an obvious indication and mindset for the user when selecting data.
- **Data Source** – SAP, Dynamics CRM, Active Directory. This level is the name of the data source itself.
- **Entity** – e.g. Table1, AQ_SAP_Customer, AQ_Outlook_AddressBook. This is the individual table or file from a specific data source.
- **Version** – e.g. v1, v2. This is the version of the schema, so if a new column is added to a file or a column is removed the automated process that picks the files up needs to be able to identify what files are using what schema, this is controlled by using the version folder structure at this level.
- **Year>Month>Day** – e.g. 2018>01>27. This is the date of which the files were added to the data lake. **Note**: this does not necessarily relate exactly to the date the row was created in the source system. This folder structure is created automatically by the data factory partitioning functions.

This structure has several benefits – it firstly acts as a metadata tagging system, we can use the folders under which a file resides to infer various attributes of the file itself. However, this becomes much more powerful when considering the tools used to process this data.

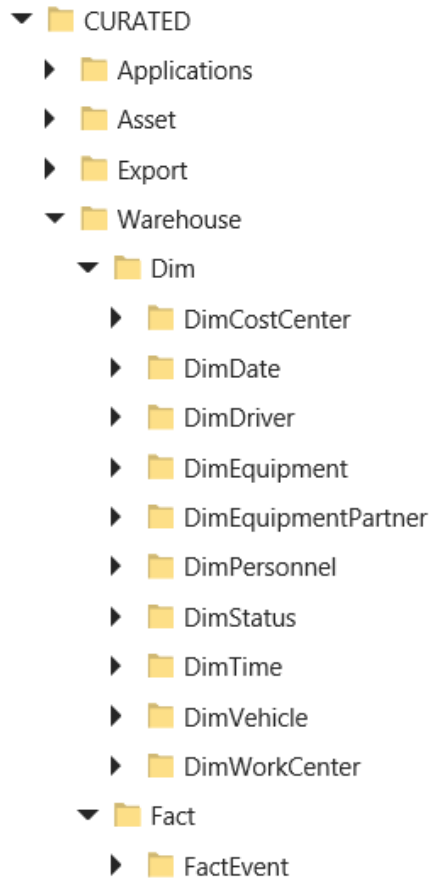
Data Lake Analytics can read files recursively from complex file structures, treating all records as a single dataset, however we cannot filter records within these files. We can, however, selectively decide to exclude files from our queries based on their parent folders and file naming.

Our folder structure recommendations are therefore optimised to assist with efficient file querying.

3.1.3 Suggested folder structure for CURATED

This area of the lake will be structured by destination data model/entity. Depending on the data set it may contain a current view snapshot of the data or it may hold a history view of the data set or maybe in some cases both.

Below is a diagram depicting the recommended structure of this area.



A description of each level is described below with examples of each level.

- **Lake Area –** e.g. Curated. This is the top of the structure.
- **Model** - e.g. Export, Warehouse, Asset. This is the data model that the data set has been created for.
- **Entity** – e.g. FactSales, DimDate, DimCustomers. This is the actual entity or data set.

3.1.4 Data Lake Store Access Model

Each individual item (folder or file) can be individually secured within ADLS. We can also set privileges on parent folder and have those permissions be inherited by child items. Access can be provided to three different types of entity:

- **AD User –** this is an individual user within Azure Active Directory. We do not directly grant individual users access to the data lake store outside of POC environments.
- **AD Group** – This is our default security choice, much like traditional on-premise systems. An Active Directory group can be created and granted access specific folders. Users can then be added/removed from this group without modifying the configuration of the lake store.
- **AD Application** – Application Registrations, also known as Service Principals act as service accounts within the Azure platform. These accounts can be directly granted access to specific folders if their use is not expected to change in the long term, otherwise they can be added to AD groups as with normal users.

Entities can be granted three different access levels, Read, Write and Execute:

Root Folder	FILE	FOLDER
Read (R)	Can read the contents of a file	Requires Read and Execute to list the contents of the folder
Write (W)	Can write or append to a file	Requires Write and Execute to create child items in a folder
Execute (X)	Does not mean anything in the context of Data Lake Store	Required to traverse the child items of a folder

Our basic security model specifies AD groups for each folder structure as follows:

Root Folder	READ	WRITE	EXECUTE	Notes
RAW	[ADLS_RAW_READ]	[ADLS_RAW_WRITE]	[ADLS_RAW_READ]	Read access provided to expert Data Analysts (i.e.: Data Scientists), Write access strictly limited to applications
BASE	[ADLS_BASE_READ]	[ADLS_BASE_WRITE]	[ADLS_BASE_READ]	Read access provided to expert Data Analysts (i.e.: Data Scientists), Write access strictly limited to applications
ENRICHED	[ADLS_ENRICHED_READ]	[ADLS_ENRICHED_WRITE]	[ADLS_ENRICHED_READ]	Read access provided to expert Data Analysts (i.e.: Data Scientists), Write access strictly limited to applications
CURATED	[ADLS_CURATED_READ]	[ADLS_CURATED_WRITE]	[ADLS_CURATED_READ]	Read access provided to a wider range of Data Analysts. This layer can be broken up into individual business areas reflected by the model (i.e.: Pollutions Mart). Write access strictly limited to applications
LABORATORY	[ADLS_LABORATORY]	[ADLS_LABORATORY]	[ADLS_LABORATORY]	Deliberately less restrictive, access to the LAB area provides more freedom for innovation and experimentation. Access can be broken down to individual projects/people if there is sensitive data being handled within the team

EXHIBIT	[ADLS_LABORATORY]	[ADLS_LABORATORY]	[ADLS_LABORATORY]	This mirrors the LAB layer to promote the sharing of completed data sets.
catalogue	[ADLS_Admin]	[ADLS_Admin]	[ADLS_Admin]	This is a system-managed folder where access should be restricted to Data Lake administrators
system	[ADLS_Admin]	[ADLS_Admin]	[ADLS_Admin]	This is a system-managed folder where access should be restricted to Data Lake administrators
AssemblyCache	[ADLS_Admin]	[ADLS_Admin]	[ADLS_Admin]	This is a system-managed folder where access should be restricted to Data Lake administrators

It is strongly advised that a more granular security design is devised during the build phase of the NGBI project. This simplistic security model demonstrates how security can be tightly controlled whilst still promoting data usage – the end requirements of the security model itself will be driven by the data held within the various areas of the store itself.

3.1.5 Use in BI Architecture

We use ADLS as the data store for all data sources for the BI Warehouse. Deltas are provided into the RAW section and then as the delta files are picked up from Azure Data Factory/U-SQL they are transformed and dropped into other sections of the Data Lake. Once in its final form, data is stored ready to be picked up by the SQL Data Warehouse and on to the reporting layer.

Traditionally with on premise systems (where data storage is expensive) our data stores were often transient, meaning we would delete the staging data once it has been loaded. Data Lake allows us to move to full delta history model, whereby we can maintain a query-able history of data loads that can be used to rehydrate the warehouse as a DR solution, or for ad-hoc queries.

Azure Data Lake allows us to host a full history delta solution of source files.

Because we are keeping a full delta history model, we need to have the ability to identify which rows have been deleted from the source system. To achieve that, we have two options:

- The source provider identifies the row was deleted by adding a IsDeleted attribute (preferable option).
- The source provider will always send the full history and we apply a Slowly Changing Dimension Type 2 approach to identify the deleted records. This option will become slower and more resource consuming as times passes.

At the time of writing, U-SQL only supports insert operations.
Independently of the selected approach, the Curated section will always contain a Type 1 (all active records) and Type 2 (all inactive records) files

3.1.6 Use in Application Integration Architecture

For the applications architecture ADLS is used very similarly as in the BI Architecture. Data will posted to a service bus from the application itself and a Logic app will move the data into its own dedicated area of the Lake, where the data will go through various transformations.

Data files can then be picked up from the Curated section of the Lake along with other data sets from the BI section of the Lake to be presented to the reporting layer.

3.1.7 Use in Data Science Architecture

The Data Lake Store has three main touch points with the Data Science Processes – Exploration, Experimentation and Publishing Datasets.

3.1.7.1 Exploration

In order to design and build models, Data Scientists need access to the relevant data. The team will be provided with access they have been approved for and will be empowered to access this in its various forms throughout the lake. Some models may source data from our curated, clean data sets, whereas others may need to access date in its purest, raw form before and data sanitisation has occurred. The team will therefore touch elements of RAW, BASE, ENRICHED and CURATED lake sections.

3.1.7.2 Experimentation

The LABORATORY area of the lake, as described in the Data Science Solution Overview, is an area of the lake where the strict governance guidelines have been loosened to remove barriers to innovation. The Data Science team can create temporary datasets, pull in their own reference data and generally use the folders structures within as a sandbox area during model development and investigation. The structure of this area is up to the Data Science team itself, we generally see two different approaches:

- Team Member "Desks" – Each Data Scientist has their own folder to maintain as they see fit. They have read access to everyone else's folders but write access to their own. This allows for different personalities, from the fastidious & rigorous data filing systems, to the "messy desk" syndrome of the creative worker.

- Project Folders – Each new Data Science project has a dated folder created. This folder is shared by any scientists working on the project and structured as they see fit. This approach makes the management and archiving of old project data more manageable, but also requires an element of project discipline.

3.1.7.3 Publishing

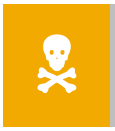
Finally, we have the EXHIBIT area of the lake. A lot of the output of Data Science teams is traditionally a machine learning model, to be integrated into other data flows and applications. However, sometimes the team performs a one-off investigation, the result of which is a dataset which holds value to the business but is not part of the automated data processing workflows.

The EXHIBIT is a hosting area for these datasets – they are intended for general business use but they have not had the same rigour and controls applied as the other assets within the CURATED area and are therefore hosted within their own specialist area of the lake.

3.1.8 Alternatives and Reasoning

3.1.8.1 Blob Storage

From a technical standpoint, ADLS uses the Web HDFS (WHDFS) implementation, whereas Blob storage is classic HDFS. The move to WHDFS changes very little about the technical implementation – the major changes being around the access API (and thus, the improved security), the removal of file size constraints (Blob storage limits at 1.4Tb per file, ADLS has no limit).



The use of PolyBase (in Azure SQDW) for data loading limits the security that can be applied to Blob storage. These limitations do not exist in Data Lake Store.

Blob storage does not have the same security concepts as ADLS. Users can mimic a folder structure within blob storage by encoding path structures into individual file names, however security cannot be granted to these virtual directories.

This means that anyone with access to the blob storage account has access to all files within that account.

In order to achieve granular control, separate blob accounts have to be created, this leads to a proliferation of accounts to mimic the folder structure that we can build natively within our Data Lake.

3.2 Azure Data Lake Analytics (ADLA)

3.2.1 Component Summary

Azure Data Lake Analytics is an on-demand analytics job service to simplify big data analytics. You can focus on writing, running, and managing jobs rather than on operating distributed infrastructure. Instead of deploying, configuring, and tuning hardware, you write queries to transform your data and extract valuable insights.

The analytics service can handle jobs of any scale instantly by setting the dial for how much power you need. You only pay for your job when it is running, making it cost-effective. The analytics service supports Azure Active Directory letting you manage access and roles, integrated with your on-premises identity system.

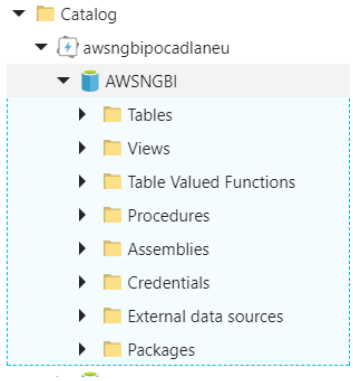
It also includes U-SQL, a language that unifies the benefits of SQL with the expressive power of C#. U-SQL's scalable distributed runtime enables you to efficiently analyse data in the Data Lake Store, across Azure SQL Database and Azure SQL Data Warehouse.

3.2.2 Data Lake Analytics Scripts & Management

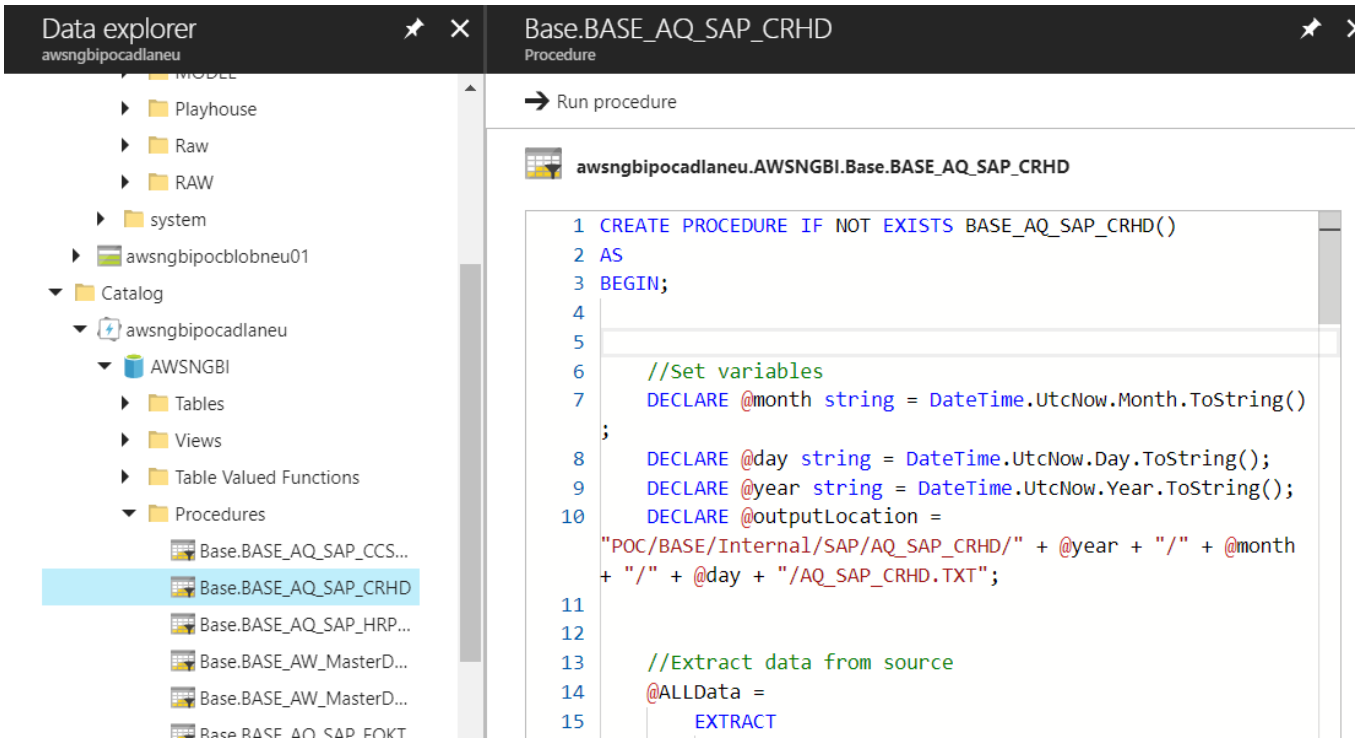
Azure Data Lake Analytics (ADLA) is an on-demand, massively scalable compute engine. Users can write a script that queries files within the Data Lake Store, performed efficient, highly parallelised aggregations & transformations and outputs files elsewhere within the store. This uses a proprietary language called U_SQL. It is important to note that ADLA does not contain any data itself, simply applying metadata & compute operations on top of storage layers such as ADLS.

As with traditional ETL systems, we are managing a code-base of transformation scripts that are used to process each individual file type. The traditional method of managing this within relational systems such as SQL Server would be to write and deploy stored procedures for each step of processing.

ADLA does contain a relational engine that is familiar in this respect – we can write stored procedures, functions and views on top our data lake directly:



We write processing scripts within Visual Studio which are added to our ADLA catalog as stored procedures. Our processing orchestration then simply needs to call the relevant stored procedure in order to perform the data processing required. These stored procedures can accept parameters and act in a very similar manner to traditional SQL Server procs.



3.2.3 Use in BI and Application Integration Architectures

In the data preparation stage of building data sets used in both BI and Applications architectures, we will be using ADLA to host and execute U-SQL procedures to transform, cleanse and create complete data sets from source data.

C# libraries will also be stored in ADLA which will be used to help cleanse data and add meta data through the loading process.

Data is not stored in ADLA, only the procedures in how to manipulate it. All data is stored inside the Data Lake.

3.2.4 Use in Data Science Architecture

The ad-hoc nature of Data Lake Analytics makes it perfect for the exploratory data manipulation that is common to the Data Science workload. The Data Science team can use U-SQL to source data from across the Lake and produce data sets within their LABORATORY areas.

They will also have the option of

They will have the option of writing their own data preparation scripts on their DSVM, however scripts written in U-SQL will have the benefit that they can easily be productionised should they wish to turn their ad-hoc data preparation into a regular curated asset.

There is also the option of extending the U-SQL language with C# or traditional data science languages such as Python or R. The team can therefore use their standard python libraries within any data preparation scripts they create.

3.2.5 Alternatives and Reasoning

3.2.5.1 Azure SQL Data Warehouse

Azure SQL Data Warehouse supports the use of Polybase which could be used to select data directly from sets of flat files, T-SQL could then be used to manipulate and transform the data and apply cleansing rules. This is a similar pattern to what is often used in on premise solutions when data is stored in separate staging databases rather than flat files. which could be used to select data directly from sets of flat files, T-SQL could then be used to manipulate and transform the data and apply cleansing rules. This is a similar pattern to what is often used in on premise solutions when data is stored in separate staging databases rather than flat files.

There are several drawbacks to this approach, the first being cost. SQL Data Warehouse is an expensive component in the solution and therefore it is recommended to only have it turned on (and therefore costing) when needed and turn off when not required. If it was being used for data movement and transformation of data in near real time it would need to be turned on at all times and would therefore incur significant costs.

The second drawback is also relating to costs and future proofing the solution. As ADLA and the other components of the solution such as Azure Analysis Services evolve, it may remove the need for having Azure SQL Data Warehouse in the solution altogether, therefore removing an expensive component within the solution. If the tools evolve in this way, we want to design the solution in such a way that removes as much dependency on SQL Data Warehouse as possible, which would then create a smooth transition into deprecating it.

Concurrency is a major issue within massively parallel processing systems such as SQLDW – even when scaled to the higher performance tiers, only 32 concurrent queries can be executing on the engine. It will naturally queue up queries, but if we are planning for an intensive file processing architecture, we would be very limited by SQLDW as our key data processor – especially if it has to cater for data processing, publishing to downstream systems and act as a sandbox for the data science team.

Our final limitation for a SQLDW-based approach is our ability to independently scale our jobs. SQLDW is scaled as a whole component and charged on an hourly basis. Unlike ADLA, where compute is allocated to individual jobs and can be fine-tuned to cater for specific intensive jobs, we would have to plan daily compute schedules for the SQLDW, which can be hard to predict.

To reduce the running costs of the solution it is recommended to turn SQL DW off when it is not required or being used. This can be automated via PowerShell and Azure Functions.

3.2.5.2 SQL Server Integration Services

SQL Server Integration Services (SSIS) packages can now be hosted in Azure and executed from Data Factory v2. SSIS was the tool of choice for data movement and manipulation, for on-premise solutions using the Microsoft SQL Stack.

The downside of using SSIS in this way is that it relies on in-memory pipelines and will only scale out within the limitations of its host server. It will not natively take advantage of distributed storage, meaning it will never be able to compare to the massive parallelisation available with modern big data processing tools, such as ADLA. Using SSIS would therefore result in a significantly slower loading process.

It is also less compatible with modern code automation tools – there are third party programs available to that can help automate the creation of SSIS packages but they tend to be overly complex and result in limited time saving. ADLA scripts are incredibly simple to generate through simple code generation tools.

SSIS is not a suitable data movement tool for large amounts of data it is limited by its host server and does not natively take advantage of HDFS and MapReduce patterns to allow large parallelisation of data processing.

3.3 Data Factory V2

3.3.1 Component Summary

The Azure Data Factory service is a fully managed service for composing data storage, processing, and movement services into streamlined, scalable, and reliable data production pipelines.

Developers can use Data Factory to transform semi-structured, unstructured and structured data from on-premises and cloud sources into trusted information. Developers build data-driven workflows (pipelines) that join, aggregate and transform data sourced from their on-premises, cloud-based and internet services, and set up complex data processing through simple JSON scripting.

The Azure Data Factory service provides monitoring and management of these pipelines at a glance with a rich visual experience offered through the Azure Preview Portal. The information produced by pipelines can be easily consumed using BI and analytics tools.

3.3.2 Use in BI Architecture & Application Integration Architecture

For the loading of the BI warehouse and Application Architecture, Data Factory is used to orchestrate and schedule the ADLA jobs which in turn perform the loading and creation of data sets.

In the BI Architecture Data Factory will also be used to obtain source files and perform the initial movement into the Data Lake.

Data from the SAP system will be loaded into Data Lake via Attunity, not Azure Data Factory.

Separate Data Factories will be used in each Architecture, which will allow each one to be orchestrated and controlled individually without having to affect the performance and operations of each.

3.3.3 Use in Data Science Architecture

There are some basic uses of Data Factory that the data science team might want to investigate – certainly basic models hosted with AML Studio can be retrained and managed via some of the inbuilt functionality of Data Factory.

However, the kind of regular, managed orchestration that data factory provides is outside the expected remit of the Data Science team. It is more likely that the team will pass these data management requests to the BICC to implement as a managed data loading pipeline.

3.3.4 Alternatives and Reasoning

3.3.4.1 Azure Data Factory v1

Azure Data Factory v1 is the current version of Data Factory with v2 currently in preview, expecting to become Generally Available in the next few months. It is not a traditional update to the current version and many of the components have been re-built from the ground up. V2 when available will hosts a lot of new functionality and will be the version where future updates will be applied as the product matures. V1 will be used for legacy systems already using it. Therefore, it is recommended for new solutions to use Data Factory V2.

Once generally available (GA) Azure Data Factory v2 will be the future of Data Factory and v1 will only be in place for legacy solutions

3.3.4.2 SQL Server Integration Services

The current implementation of SSIS within Azure requires the packages to be executed from within Data Factory V2. Whilst SSIS packages do contain more sophisticated orchestration / control workflow functionality, these features would be of limited use given all scheduling and execution is controlled by the parent Data Factory.

If we were using SSIS for data movement and manipulation, we might utilise elements of the control flow for further processing logic within the package. However, we have ruled SSIS out as our data processer (see section 3.2) which means we can also rule it out as our orchestration tool.

3.4 Azure SQLDB

3.4.1 Component Summary

SQL Database is a general-purpose relational database service in Microsoft Azure that supports structures such as relational data, JSON, spatial, and XML. It delivers dynamically scalable performance and provides options such as column store indexes for extreme analytic analysis and reporting, and in-memory OLTP for extreme transactional processing. Microsoft handles all patching and updating of the SQL code base seamlessly and abstracts away all management of the underlying infrastructure.

3.4.2 Use in BI Architecture & Application Integration Architecture

In BI and Applications Architecture, the Azure SQL Database serves two purposes, as a metadata store and hosts logging information. These are descried in more detail below:

- **Metadata Store** – Data around the files/tables and columns including data type and names etc. is stored in the metadata store. From this metadata, we can automatically generate scripts and loading patterns to produce many of repetitive tasks in the loading procedures.
- **Logging Database** – Used to store logging information of the movement of data and load progress for the entire solution.

3.4.2.1 Solution Logging Methodology

The logging approach proposed in this architecture is designed to be dynamic and support the logging of all components used, present and future. Creating/Updating a log entry can be done by calling a SQL Stored Procedure this makes it accessible to as many of components as possible.

All logs are then stored inside the SQLDB instance, this has been chosen to make the logs accessible and easily querable for different people and applications.

3.4.3 Use in Data Science Architecture

There are occasions once data has been collected where the analysis of this data is more easily achieved in a standard SQL environment. This has not been provisioned for within the architecture, under the assumption that SQL can be used for data interrogation within Jupyter notebooks hosted on the Data Science VMs, they therefore should not need a separate SQL instance to manage data long term. If a future project requires a lightweight SQLDB to be provisioned, this can easily be hosted within the same logical SQL Server as the orchestration & auditing SQLDBs.

There is a possibility that in future projects, the data science team may want to perform deep analysis on the logs generated by the NGBI platform itself. If this requirement arises, access will be provisioned via the standard processes.

3.4.4 Alternatives and Reasoning

3.4.4.1 Cosmos DB

CosmosDB is a hugely scalable document store, used to manage large collections of data using a set of common NoSQL database formats. The document store flavour of Cosmos DB would be a viable solution for the meta data and log stores but does not give a real benefit over a SQL database given our workload requirements, and is not as easy to query for ad-hoc checking and monitoring.

We have used CosmosDB in many solutions where the metadata requirements were much more varied and required the flexibility granted by JSON documents over relational tables.

If Cosmos DB was used elsewhere in the solution it may have been selected to also host the metadata store and log details. But as neither SQL DB nor Cosmos DB were being used and a new component was required either way SQL DB has been selected as we can implement the Adatis design pattern to accelerate the design of this structure.

3.4.4.2 Azure Data Lake Store

The logs and metadata could be stored in flat files inside the data lake, but the data lake is optimised for large batch data processing and aggregation. The lack of indexing within files means that transactional workloads, such as those required within logging and control systems, would be very inefficient and slow. The ADLA catalogue tables would be a closer match, however the access limitations of these tables mean that we cannot use them outside of ADLA itself, they are therefore not a viable storage location.

3.5 Azure SQLDW

3.5.1 Component Summary

SQL Data Warehouse is a cloud-based Enterprise Data Warehouse (EDW) that leverages Massively Parallel Processing (MPP) to quickly run complex queries across petabytes of data. Use SQL Data Warehouse as a key component of a big data solution. Import big data into SQL Data Warehouse with simple PolyBase T-SQL queries, and then use the power of MPP to run high-performance analytics.

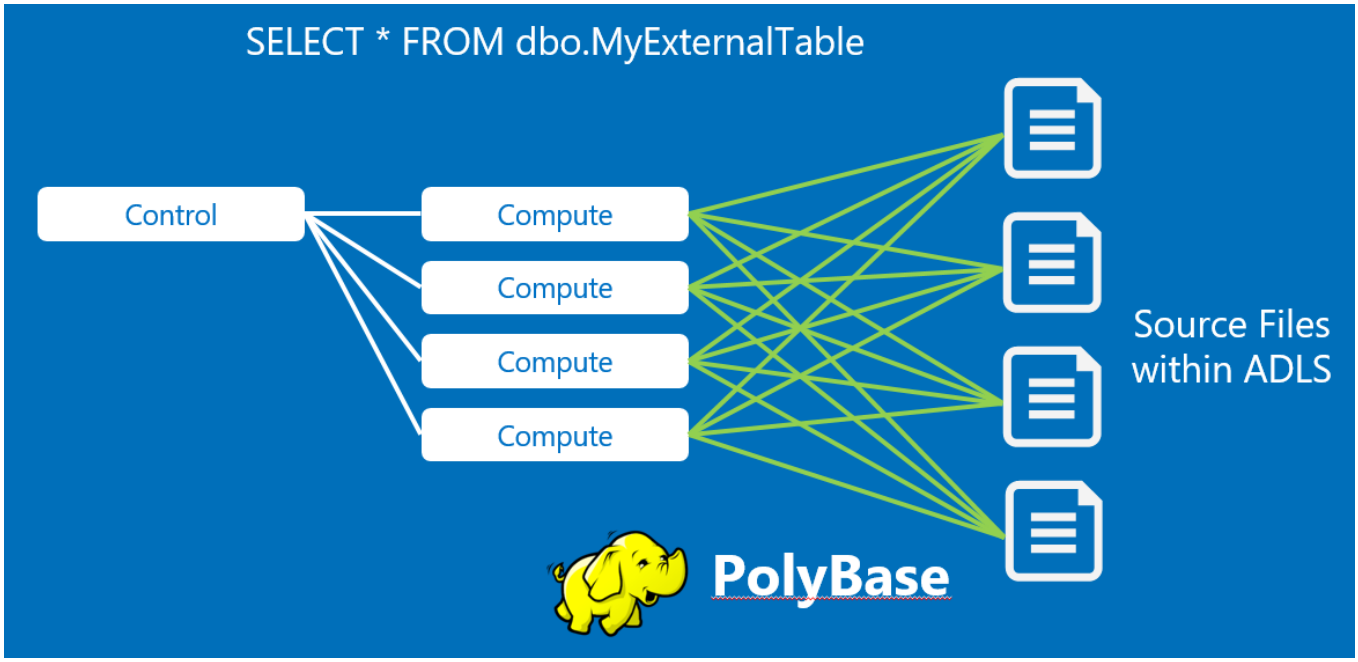
3.5.2 SQLDW & PolyBase Data Loading

Azure SQL Data Warehouse (SQLDW) is a specialist massively parallel processing (MPP) implementation of the SQL Server engine. The heavy reliance on parallelism within the engine means it can process vast amounts of data very quickly, but this also means the engine internals work differently to a standard SQL Server.

There are various ways that SQLDW can be utilised within a modern analytics platform. It can be used in place of Data Lake Store where a more relational data storage layer is desired, although the costs and patterns are quite different.

However, it is also very powerful as an interactive querying layer over the flat file stores held within our Data Lake store. The SQL Server feature known as PolyBase implements a system of parallel Hadoop File System Bridges – essentially we can run many flat file data readers in parallel, all controlled via T-SQL.

If we have users who are skilled in traditional SQL querying, they can use the SQLDW as a querying layer. When each query is executed against PolyBase tables, the query will read from files within the ADLS rather than within SQL tables.



This combination of technology means we can integrate our flat-file layer within Data Lake Store with any technology that can query a traditional SQL data store.

3.5.3 Use in BI Architecture & Application Integration Architecture

For the BI and Applications Architecture, Azure SQLDW will be used to create a logical representation of the data to present to the Azure Analysis Services.

The data will be stored inside the Data Lake in the form of flat files, and SQL DW will use PolyBase to ingest the files and present them as tables to Analysis Services.

3.5.4 Use in Data Science Architecture

It is unlikely that the Data Science team will make heavy use of the SQLDW platform itself unless working in a language that cannot natively access flatfile data. The python projects evaluated thus far would be able to access the ADLS store directly and not require the SQLDW intermediary layer.

Should access be required in future, or specific lookup/reference data start to be curated within the SQLDW itself, then access will be provisioned as required.

3.5.5 Alternatives and Reasoning

3.5.5.1 Azure SQL Database

Azure SQL Database could be used to store a physical version of the warehouse to present to Analysis Services. Azure SQL database does not support the use of PolyBase to ingest files from the Data Lake, therefore additional Data Factory jobs would be required for the data movement and then a physical copy of the warehouse would exist which would then incur more storage costs on the database.

Performance of loading large amounts of data is a key consideration, and PolyBase in SQL DW will provide superior performance over Azure SQL DB.

3.5.5.2 None

As part of the PoC we will also be investigating if it is appropriate to remove the use SQL DW and the implications that this may have, if Azure Analysis Services (AAS) pulls data directly from Data Lake Store. This has a direct impact on how we might structure files held within the data lake store, as we would want to implement partitioning within the AAS structure for efficient model processing – SQLDW would allow us to use the traditional time-bound search predicates seen in many solution designs.

We are building this architecture on the assumption that the technology landscape will change. We therefore expect that whilst having no querying layer between AAS and ADLS is not the most efficient solution now, it may well become more optimised in future. When this happens, we can remove SQLDW from the architecture without drastic changes to the solution.

3.6 Azure Analysis Services

3.6.1 Component Summary

Azure Analysis Services provides enterprise-grade data modeling in the cloud. It is a fully managed platform as a service (PaaS), integrated with Azure data platform services.

With Analysis Services, you can mashup and combine data from multiple sources, define metrics, and secure your data in a single, trusted semantic data model. The data model provides an easier and faster way for your users to browse massive amounts of data with client applications like Power BI and Excel.

3.6.2 Use in BI Architecture & Application Integration Architecture

Analysis Services will be used as the semantic layer for the BI and Applications Architecture. It will host data models in the form Tabular Models which will contain dimensions as business entities and hierarchies, and

measures as KPI's and metrics. The Tabular models are presented in a business focused fashion, to allow non-technical users the ability to browse and understand the data and gain the insight they require.

The structure and quantity of AAS models that will be required will be determined on a case by case basis as new applications or business areas are added to the architecture.

Larger business areas (containing many data sets) should be stored in their own model as the model can then be scaled and managed independently due to the requirements of the data set.

For smaller data sets/business areas such as many of the ones that follow the Applications Integration Architecture should be grouped together, based on sharing similar management requirements. This will reduce the number of models required and therefore reduce the running costs of Analysis Services.

Perspectives and security can be placed on tables and data sets to ensure users only have access to the correct data sets, even if their data set is in a shared Analysis Services model.

3.6.3 Use in Data Science Architecture

The traditional model-building activities of the data science team will not be sourcing data directly from the analysis services model, unless there are specific DAX calculations that they require the output of.

Analysis Services could present some uses for the data science team should they want to share the output of their findings with the business – however as this would only represent a tactical solution it would be more prudent to source the data directly into PowerBI. If the PowerBI model then needs to be productionised, the dataset would be brought into the formal ETL routines and follow the standard data loading process of the NGBI platform.

3.6.4 Alternatives and Reasoning

3.6.4.1 Power BI Data Sets

Power BI has the ability to create and host Tabular models (labeled Power BI Data Sets) inside the Power BI service. There are limitations around size and complexity with the models that it can create.

It is therefore not suitable for "Big Data" projects as the model size will exceed the Power BI data set limit of 10GB. Partitional and data loading are also restricted within Power BI data sets among other features that will be required for a solution of this complexity.

3.7 Power BI

3.7.1 Component Summary

Power BI is a suite of business analytics tools that deliver insights throughout your organization. Connect to hundreds of data sources, simplify data prep, and drive ad hoc analysis. Produce beautiful reports, then publish them for your organization to consume on the web and across mobile devices. Everyone can create personalized dashboards with a unique, 360-degree view of their business. And scale across the enterprise, with governance and security built-in.

3.7.2 What is Power BI Premium vs Power BI Pro

Power Bi Premium is a capacity based licensing model as opposed to the traditional per user license.

The Premium option will create dedicated capacity within the Power BI Service for an organizations tenancy. This allows for dedicated scaling and performance management.

Reports hosted in Premium capacity will be accessible to non-licensed users, making them shareable inside and outside of the company without the user requiring a Power BI Pro license.

Free users connecting to Power BI Premium capacity can consume and interact with pre-built reports, via Power BI Apps or Content Packs.

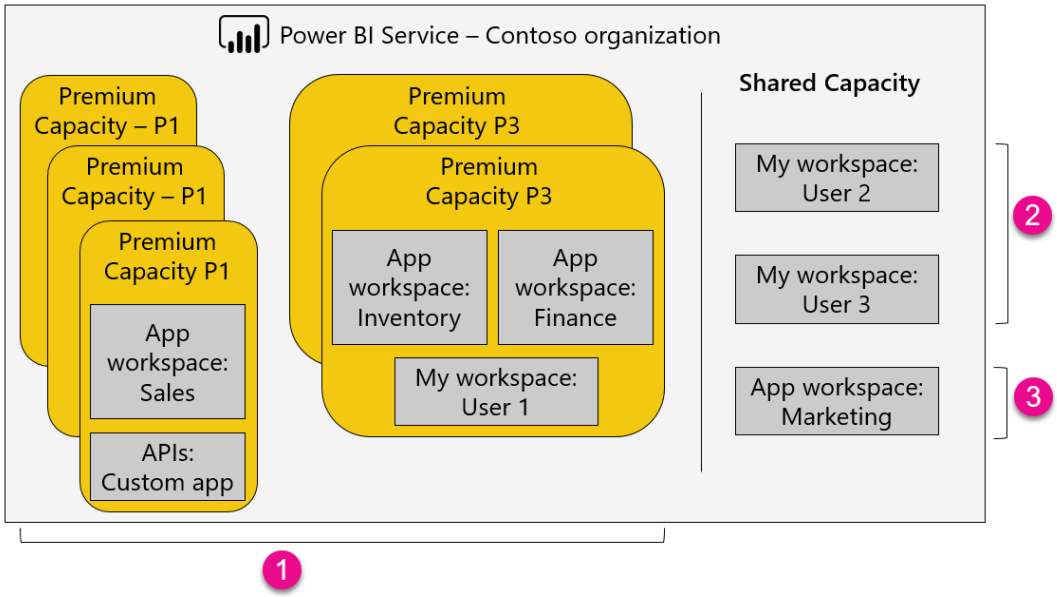
- A Pro license is required to create content for Power BI Premium
- A Pro license is required to share content with users within Premium
- A Pro license is required for Ad-Hoc analysis in Power BI Premium
- A Pro license is not required to consume content, created by a user with a Pro license.

Power BI Premiums use within AWS has not yet been defined and will be defined on case by case basis as projects are built/migrated to this architecture. The main driver for reports being placed in Premium Capacity should be around if the report is to be view by large numbers of users without them requiring the ability to self-serve on the data set behind it and share that self-serve content.

The sharing of self-service content is only available to users with a Pro license, therefore reports that require users this ability should *probably not be placed in Premium Capacity. (*there are other drivers that may cause the need for Premium Capacity)

Power BI Premium is not necessarily an Enterprise version of Power BI. It introduces enterprise capability but should be combined with Power BI Pro Licenses based on the requirements, it is not one or the other.

The diagram below outlines an example of how Premium capacity may be used in conjunction with the traditional shared capacity.



1. Items within a Premium capacity

- Content readers can be Free or Pro users.
- Sharing requires a Pro license.
- Content building requires a Pro License

- REST APIs for embedding utilise a service account, with a Pro license, rather than a user.

2. My workspace in Shared capacity

- Content build can be done by all users (Free and Pro)
- Sharing requires a Pro license.
- Accessing shared content requires a Pro License

3. App workspaces in Shared capacity

- Any app usage requires Pro licenses

3.7.3 Power BI Embedded

With the release of Power BI Premium, a new REST API for Power BI is also available. The new API will allow Power BI reports to be embedded into a web page via an iFrame.

Power BI Embedded is authenticated via a token and therefore does not require a Power BI license for end users.

This results in the ability to embed Power BI reports into externally facing web pages, and embedded content into internal applications that support the use of embedded content via an iFrame.

When using Power BI Embedded, security for the content is then pushed to the hosting application, controlled via which users have access to the web page.

3.7.4 Power BI Connection Type Import mode VS Live Connection

There are two main connection types to data inside Power BI:

- Import mode
- Live Connection / Direct Query (Dependent on whether connecting to Analysis Services or a Database)

3.7.4.1 Import Mode

When using the Import Mode connection inside Power BI the data used in the report is loaded directly into the Power BI service and is then hosted from this location for report use.

It allows users to combine data sets from various locations, i.e. users can draw some data from a database and some from other flat files loaded in, join the data sets together and create reports on them.

Whilst this functionality is excellent and truly useful in some situations it should not be the primary connection type when using Power BI.

There are some limitations of using Import Mode, these can be found below:

- Data sets cannot exceed 1GB (this limit is increased when using Power BI Premium and based on size of the capacity).
- Ownership of the data set is given to the report builder and not stored in a central location.
- Data set quality is in the control of the report builder and can no longer be administered by the BI team.
- Increased strain on the Power BI capacity (shared or Premium). All processing and aggregating of data in done inside Power BI Capacity, and with high report usage may cause performance issues.

3.7.4.2 Live Connection

When using Live Connection inside Power BI the source data remains inside the source system. And when reports run a query is send to the source, calculations and aggregation is carried out on the source system and only the result set is returned to Power BI.

This allows users to create reports on much larger data sets as all the data does not need to be loaded into Power BI, or the user's development machines memory.

Having all aggregations and calculations carried out inside Analysis Services also allows much finer control over performance as the Analysis Services server can be scaled to meet the performance needs.

Live Connection should be used as much as possible as it means that data sets remain administered and controlled by the BI team and not by self-service end users.

3.7.5 Use in BI Architecture & Application Integration Architecture

Power BI will be used as the Enterprise reporting tool, it will connect to the Analysis Services Tabular model where pre-built reports from the BI development team will be available.

Power BI is an industry leading self-service reporting tool and will also be utilised for end users to connect to the Tabular model and create their own reports.

Power BI also has fantastic support for real time streaming data sets, where reports can connect directly to Azure Streaming Analytics to display data and reports in real time.

3.8 Logic Apps

3.8.1 Component Summary

Azure Logic Apps simplifies how you build automated scalable workflows that integrate apps and data across cloud services and on-premises systems.

Every logic app workflow starts with a trigger, which fires when a specific event happens, or when new available data meets specific criteria.

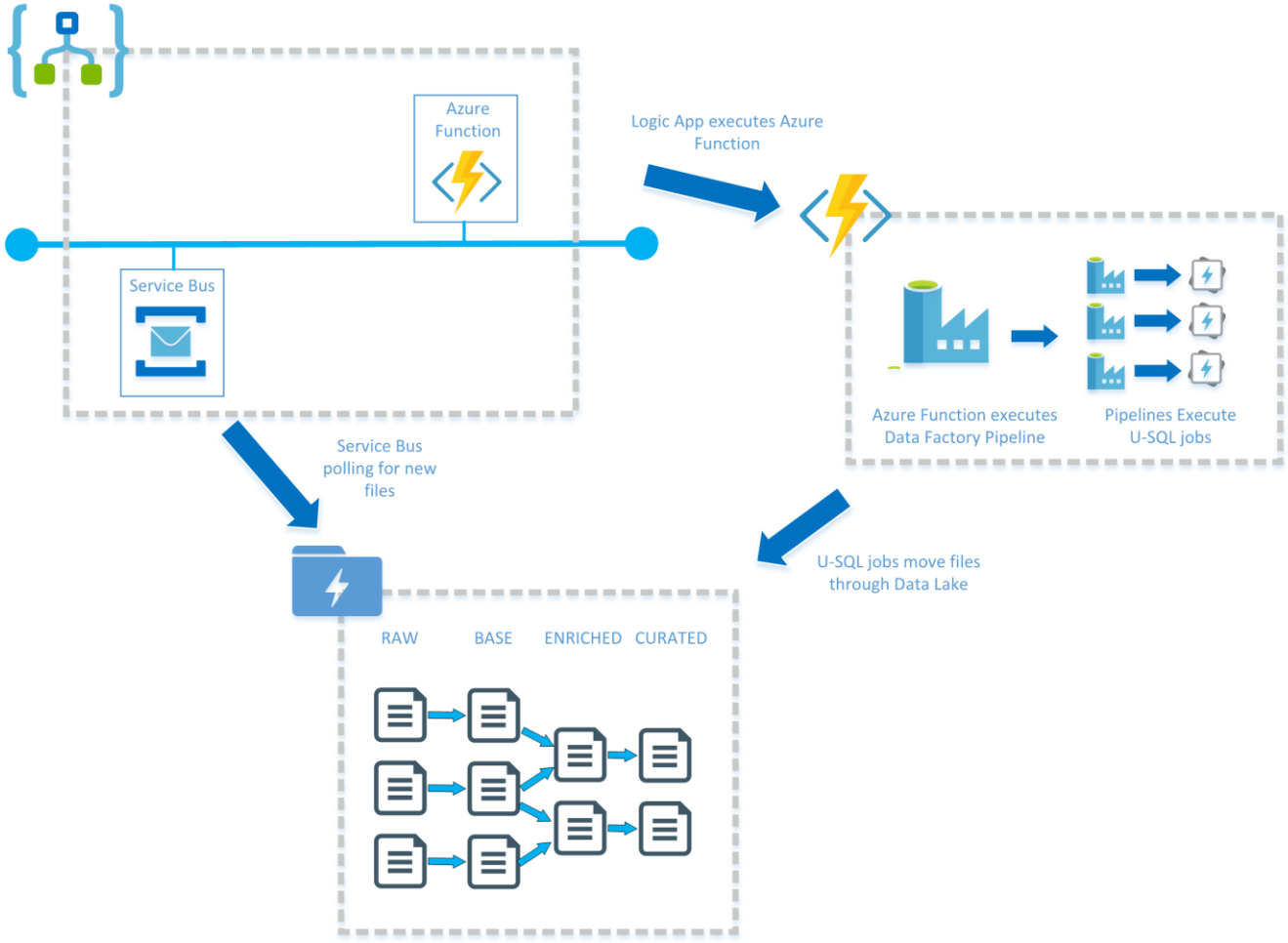
Each time that the trigger fires, the Logic Apps engine creates a logic app instance that runs the workflow's actions. These actions can also include data conversions and flow controls, such as conditional statements, switch statements, loops, and branching.

3.8.2 Event Driven Loading methodology

Traditionally in when loading Data Warehouse, we have a nightly processing window which would be based on a schedule that would execute data movement at a specific time, often loading from many different data sources in parallel.

The drawback to this approach is downstream activities cannot execute fully upstream data is available, resulting in long processing windows and if failure occurs in one part the entire process is stopped.

With the modern cloud technologies available we are able to take a more efficient approach. An Event driven loading approach is what is suggested in this architecture, whereby datasets are created and loaded as soon as all dependant datasets are available. This is depicted in the diagram below.



This removes the long processing window and as each dataset is independent of others and not orchestrated by a single process datasets will be refreshed when its source data is available and is not dependant on other datasets loading successfully.

This is implemented in the form of a service bus within a Logic App, which polls folders in ADLS to see if files are available, and if they are, then using an Azure function, also within the Logic App, and ADF pipelines to load data into the next stage of the process.

3.8.3 Use in Application Integration Architecture

Logic Apps are being utilised in the Application Architecture to monitor messages that are posted to an Azure Service Bus. When a message (data) is posted to the bus, the Logic App will then move the data to the dedicated area in the Data Lake for the application. The Logic App will then trigger a Data Factory pipeline which will perform the ETL and data movement logic required for the data, and leave it in its final form in the curated section of the Data Lake, ready to be picked up by the reporting layer.

Logic Apps will also be utilised to execute the Azure Automation scripts which will in turn process the Analysis Services models.

Azure Automation scripts can only be scheduled to run at a minimum frequency of 1 hour. Executing Azure Automation scripts via Logic Apps allows more frequent executions for near real-time reporting.

3.9 Azure Functions

3.9.1 Component Summary

Azure Functions is a solution for easily running small pieces of code, or "functions," in the cloud. You can write just the code you need for the problem at hand, without worrying about a whole application or the infrastructure to run it. Functions can make development even more productive, and you can use your development language of choice, such as C#, F#, Node.js, Java, or PHP. Pay only for the time your code runs and trust Azure to scale as needed. Azure Functions lets you develop server-less applications on Microsoft Azure.

3.9.2 Use in Application Integration Architecture

Azure Functions allow us to trigger and control the execution of Data Factory Pipelines. In the Applications Architecture, we aim to have near real-time reporting and therefore the Data Factory Pipelines do not suit being on a fixed schedule. Utilising Logic Apps and Azure Functions allows us to control their execution at a more granular level.

3.10 Azure Automation

3.10.1 Component Summary

Azure Automation delivers a cloud-based automation and configuration service that provides consistent management across your Azure and non-Azure environments. It consists of process automation, update management, and configuration features. Azure Automation provides complete control during deployment, operations, and decommissioning of workloads and resources.

3.10.2 Use in BI Architecture & Application Integration Architecture

Azure Automation will be utilised to process (update) the Analysis Services models. Depending on the model and its requirements the Azure Automation scripts will use the built-in schedule to execute this. If the model requires to be updated more frequently than each hour, then a Logic App will be used to executer the Azure Automation script.

3.11 Data Science Virtual Machine

3.11.1 Component Summary

The Data Science Virtual Machine (DSVM) is a customized VM image on Microsoft's Azure cloud built specifically for doing data science. It has many popular data science and other tools pre-installed and pre-configured to jump-start building intelligent applications for advanced analytics. It is available on Windows Server and on Linux.

3.11.2 Use in Data Science Architecture

The Data Science VM will be used as a development and exploration box, for the Data Science team to explore the data, create data sets, set up and initially test models. The machine comes pre-installed with the Azure Command Line-Interface (CLI) toolset, which is key to automating the creation of containers where machine learning models will be hosted.

The machine will host a set of different tools to help the team connect to data, manipulate data and carry out their exploration and it is assumed that the majority of model development work will be carried out on these machines.

3.11.3 Alternatives and Reasoning

3.11.3.1 On Premise Development Server

Due to the wide variety of tasks that this server will carryout there is no direct comparison to a direct equivalent. Its entire purpose is to promote easy exploration and trialling of different methods and models over the data and therefore a virtual machine is the best tool for the job.

The limitation of an on-premise VM is that it is not as easy to scale and add more compute power for when working with large datasets.

3.12 Azure Machine Learning (AML) Services

3.12.1 Component Summary

AML Services is a suite of Azure products and tools designed to simplify the process of developing, testing and productionising Machine Learning Models, without compromising the flexibility and extensibility offered by the python programming language.

The tools within the suite are:

- Machine Learning WorkBench
- AML Experimentation Services
- AML Model Management Services

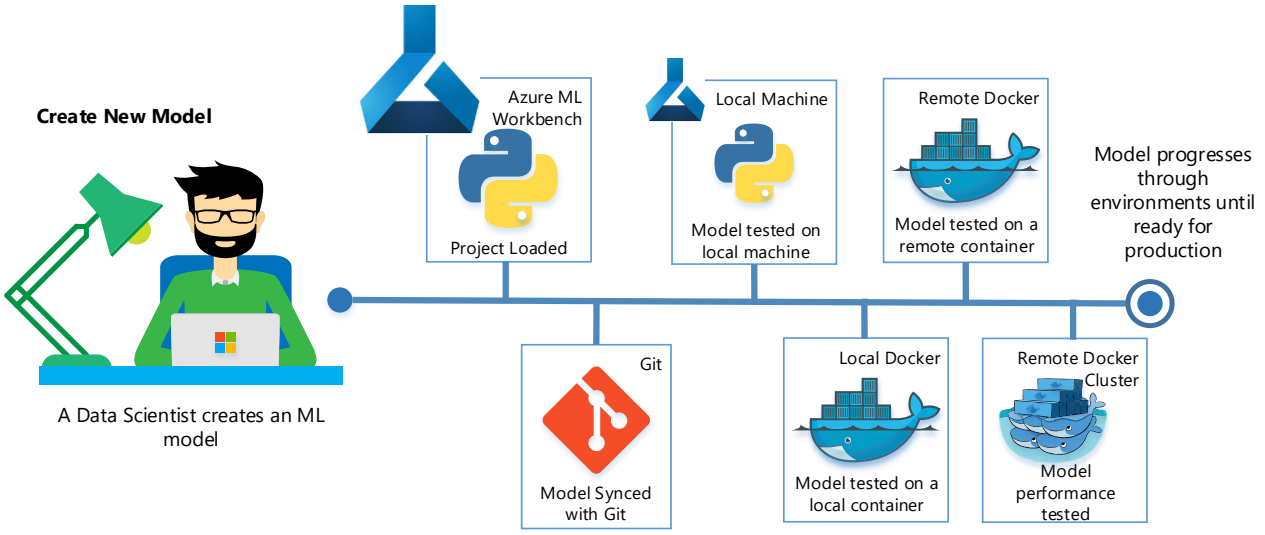
3.12.2 Development Process

Machine Learning Workbench is a client application for managing Machine Learning projects. It contains functionality for performing data preparation routines and linking them into python scripts. The actual python development requires a separate IDE, it is assumed that VSCode or another lightweight editor will be used for this. Workbench includes integration with a VSTS Git repository, allowing for full model management during the development lifecycle.

Whilst a model is being tested, the **Machine Learning Experimentation Service** can be used to describe and provision runtime environments on the fly. This means that our model can be run in different environments to test compatibility, this normally runs in the following order:

- Local – Run the script on the current client machine, using all available libraries and dependencies. This ensures that the code is functioning properly, but does not guarantee that the script contains provisioning for all libraries required. This is the first step of checking the model output
- Local Docker – This will provision a new docker container on the location machine, this isolates the code from the rest of the client operating system, ensuring that the packaged model contains all scripts and libraries required to execute. This script is still limited by the available resources on the client machine.
- Remote Docker – We can now completely remove our client machine from the equation, deploying the code to a temporary container on a remote machine, providing more confidence in the deployment manifest
- Remote Cluster - Finally, we can deploy the model to a remote cluster, where we can test performance and scalability without the limitations of our development VM.

Experimentation Process:



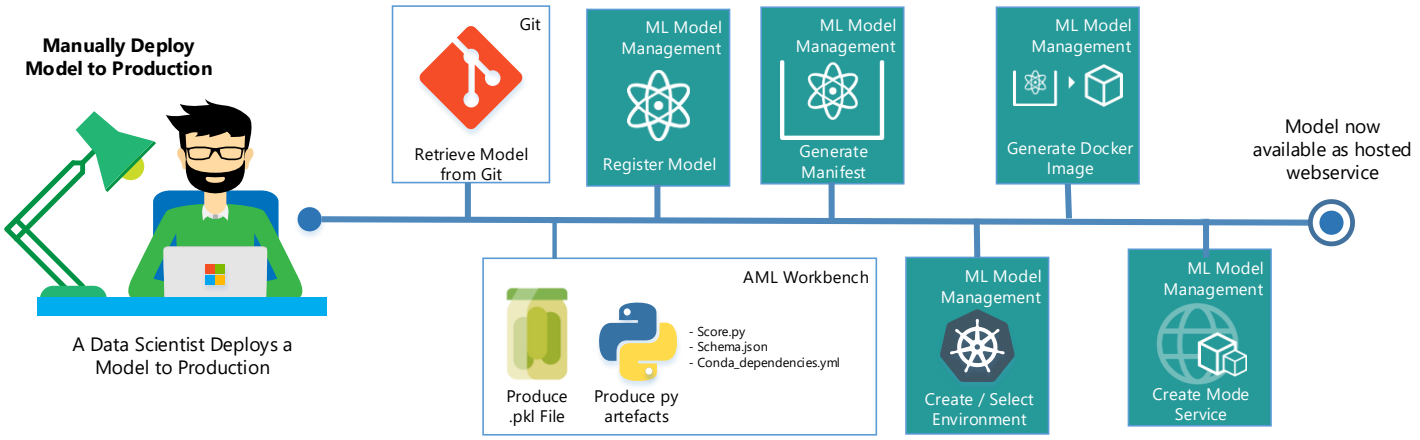
The experimentation service provides reporting on the results and outputs over various runs to compare environments and settings as the model nears release state.

Once a model is performing and has been sufficiently tested, we can push it into the production pipeline.

3.12.2.1 Interactive Model Productionisation

The user of docker and containers has now grown to be the standard approach for hosting machine learning models in a production environment. However, the creation, provisioning and management of these containers can grow into a complicated, time-consuming endeavor as the model estate grows in size.

AML Model Management Services aims to address this issue by providing a managed environment for hosting models on a shared, elastic environment. The service will provision and maintain docker clusters, using Kubernetes as a load management interface to scale elastically.



Once the model has been tested and deemed to be accurate, the model must be serialised into a pkl file. This is a serialised form of the model that can be restored to other environments.

The AML MMS expects certain objects in order for the Model Management's API and Telemetry services to properly connect to the hosted model. The required objects are:

- score.py – a script which has two mandatory functions. An init function which loads the model from the registered pkl file, and a run function, which runs the supplied input parameters against the model and returns a prediction.
- schema.json – This file is optional and is used to validate input/output objects, as well as automatically define the swagger document for the eventual web-service wrapper.
- conda_dependencies.yml – If there are additional required python packages, these can be described in a conda dependencies file

The pkl file is then registered with the Model Management Service. The registered model, along with the scoring file and other assets, are unpacked to produce a Manifest file of the various dependencies required in the eventual model.

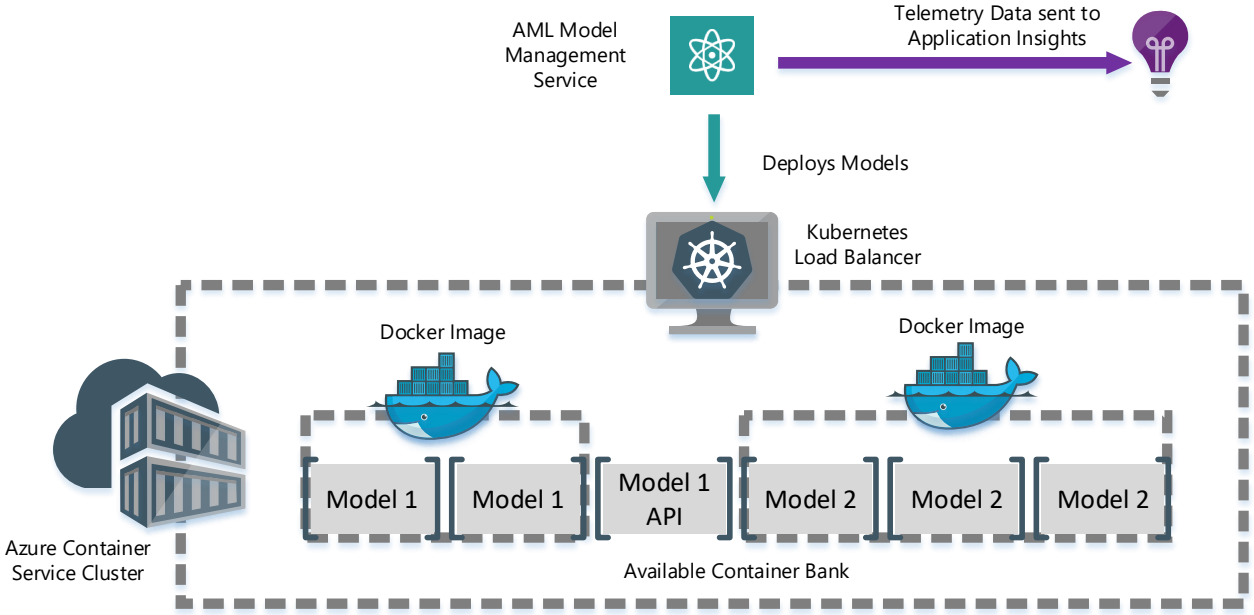
Once the manifest and model are successfully registered, a target environment is selected. If there is no existing target environment, one can be provisioned – this creates a new instance of the Azure Container Services, with Kubernetes as a load balancer & scaling management system overhead.

All of the items required for the container service are created on a dedicated resource group in the background, but the allocation of containers, along with the sizing and usage of these container environments is managed through the AML Model Management Surface.

With an environment established, the AML MMS will then create a docker image using the target container cluster's details. This image will be checked, validated but not yet deployed to the cluster.

Finally, a new Service can be created. This deploys the docker image along with auto-scaling configuration, which defines how many replica images to create and the scaling thresholds. This service provides the API wrapper around the model and can also send telemetry data to Application Insights.

Our overall architecture is therefore:



This architecture will support the majority of machine learning models as a range of dependencies can be packaged up. However, if the model requires different versions of the python language, or has to be written in an alternative language, we would repeat the above process but instead manually deploy the model to

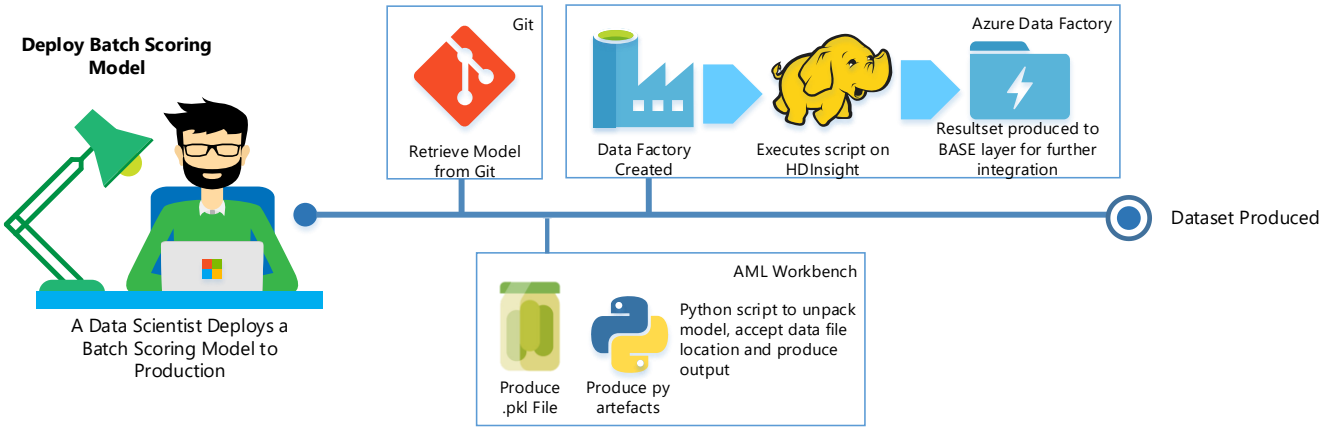
fixed Docker containers, outside of the Azure Machine Learning Services environment. As this is the exception case, it has not been fully designed as a part of this HLD.

3.12.2.2 Batch Model Productionisation

The Batch Model development process follows the same steps as the interactive model development and so they have been left as a single process.

Batch models, however, might not require the same level of API wrapping as interactive ML models, and so may not be appropriate for the full AML MMS deployment process. If there is a highly compute-intensive process that needs to run across very large datasets, we may want to move to alternative technologies such as Spark.

The model would still be written as a python model, however the eventual deployment destination would be a HDInsight Cluster. We use a generic orchestration factory that calls a PY script – this script loads the specific pkl model from our script library and passes the dataset location as an input, along with the output destination. This can then be integrated into the wider ETL process.



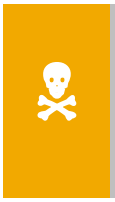
Whilst this process is not a firm requirement of the data science team right now, it is assumed that the requirement will be encountered in future projects and so the process is described here.

> At the time of writing, Spark on HDInsight is the only secure, reliable method of hosting Spark in the Azure platform. An alternative Azure component called Azure DataBricks is expected to be released mid-2018. Once released and fully Active-Directory integrated, this will be a cheaper and more flexible platform for hosting batch models.

3.12.3 Use in BI Architecture

The main usage of the above architecture in the BI workflows would be to server as dataset production once the batch requirements are known. The batch workflow would produce datasets that are then picked up by the enrichment and curation data ingestion processes to be included in the final curated datasets.

The reporting layer can also perform record lookups against the interactive layer to augment query results in real-time, provided it is only used against smaller, aggregated datasets.



It is not recommended to utilize interactive models directly from the big-data processing engines utlised in the Data Ingestion processes. The model hosting approach assumes a record-by-record scoring process, if this is attempted across a large dataset (1 million+ rows), it could easily cause a DDOS attach on the hosting layer

3.12.4 Use in Application Integration Architecture

This architecture is designed to abstract other applications from the complexities of model management. The deployed models will be surfaced via an auto-scaled, load-balanced API that they can access like any other web API.

3.12.5 Use in Data Science

This part of the architecture exists solely to service the Data Science team and will be heavily used in their day-to-day processes, which are described above.

3.13 Data Catalog

3.13.1 Component Summary

Azure Data Catalog is a fully managed cloud service that provides any user (analyst, data scientist or data developer) with the capability to discover, understand and consume data sources available in the organization. The crowdsourcing model of metadata and annotations provides the organization with the capability to 1) enrich the data assets registered in the catalog at any time; 2) let users easily understand the purpose of the data and how it is being used within the business; 3) allow the IT department to maintain the control and oversight over all the data sources.

3.13.2 Use in all Architectures

Azure Data Catalog will be used to build an inventory of all the available enterprise data sources. It will be a single and central place where all the users can categorize, tag and annotate the contents and therefore help the organization to be GDPR compliant.

4 Solution Components Management

4.1 Azure Data Lake Store (ADLS)

4.1.1 Monitoring

The ADLS portal blade provides basic information around total storage, costs incurred, read/write requests and data ingress/egress. These are the crucial factors that will determine spend and can therefore be used as the basic monitoring solution. As the Store matures, a monitoring solution that enforces metadata capture & governance should be implemented.

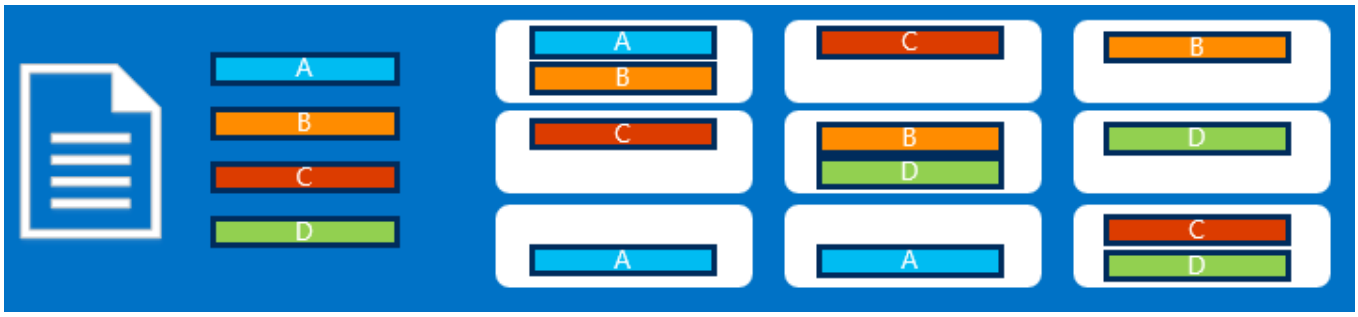
These metrics are available via the Operations Management Suite and can be funnelled into a generic monitoring solution should it be required.

4.1.2 Performance and Scaling

The most common performance bottleneck in traditional data processing systems is caused by I/O on the storage layer. HDFS systems are specifically designed to counter this performance bottleneck by distributing small parts of files (extents) across multiple cheap disks. Rather than accessing a single file on a disk, we can instead access many parts of the file at once.

On-Premise HDFS systems must be managed – files need to be distributed and disks need to be cycled and upgraded over time. If the system begins to run out of space, further disk arrays have to be added and files distributed over the new array.

ADLS, however, manages this for you. The component will automatically decide how to distribute files that are updated over a cluster of disks, including redundancy and fault tolerance. Storage scales dynamically with no need for manual intervention. For example, a file with four extents, would be distributed as follows:



As files are loaded to the Data Lake Store, it will seamlessly review the current capacity against disk usage and, if storage is running low, preemptively allocate further disks and redistribute the data. This is normally a manual job when managing an on-premise HDFS cluster and is an essential part of why ADLS can scale to many petabytes of storage.

4.1.3 High Availability & Disaster Recovery

4.1.3.1 Hardware Failure

Similar to blob storage, ADLS employs automated local replicas. Any transient hardware failure will automatically failover and be invisible to the applications performing data requests.

Data within the Data Lake Store is resilient to hardware failures within a given region as standard automated replicas are made of all files held within.

4.1.3.2 Region Wide Outage

There is no geo-replication functionality built into ADLS at this point – this means that the data is only ever held within the data centre chosen. There are three main options available in terms of creating a Geo-Redundant Highly Available Data Lake Store, each with varying degrees of availability and in turn costs associated.

Option	Coverage	Cost	Notes
None	Low	Low (nothing)	Geo-Redundancy has been planned for release in ADLS in "CY2028".
Replicate part of ADLS	Medium	Medium (~50%)	A separate ADLS could be set up in a different region. ADF could then be used to copy data from just the CURATED section of the Data Lake to the alternative ADLS. This would provide a replica of the final curated data sets.
Replicate entire ADLS	High	High (~120%+)	A separate ADLS could be set up in a different region. ADF could then be used to copy data from the Data Lake to the alternative ADLS. This would provide a replica of the final curated data sets, as well as all RAW (source) data that went into creating all data sets stored within the Data Lake.

4.1.4 Patches & Upgrades

The store itself is a very lightweight application and will encounter very few changes. However, these changes will be managed and rolled out automatically as with other Platform as a Service offerings within Azure.

4.1.5 DevOps

Adatis has produced a lightweight tool to manage folder structure creation and security allocation. This PowerShell script will read a JSON structure to understand the desired root folders and apply the security provision at each level.

The allocation of security to existing folders can be a very slow process once the data lake is up and running as it performs a minor metadata change on every single file within a root structure. It is therefore advised that this folder creation script is only used when first provisioning a new Data Lake environment.

4.1.6 Security

Security is managed via Azure Active Directory, individual users can be granted Read, Write or Execute permissions on individual securables (folders of files). The default Adatis security structure has been provisioned for the POC, as discussed in Data Lake Storage section.

The Store can also be locked down to specific IP ranges using the Firewall, giving a much more comprehensive security solution than blob storage, which cannot be locked down in such a manner. The POC will deliver a standard framework for structuring and the operational management of folders and permissions going forward.

Data Lake Store also provides encryption for data that is stored in the account. You can choose to have your data encrypted or opt for no encryption. If you opt in for encryption, data stored in Data Lake Store is

encrypted prior to storing on persistent media. In such a case, Data Lake Store automatically encrypts data prior to persisting and decrypts data prior to retrieval, so it is completely transparent to the client accessing the data. There is no code change required on the client side to encrypt/decrypt data.

The storage will therefore be:

- Encrypted at Rest
- Securable to a more granular level
- Managed via Azure Active Directory, specifically via Service Principals (aka: Service Accounts)
- Behind a configured PaaS Firewall

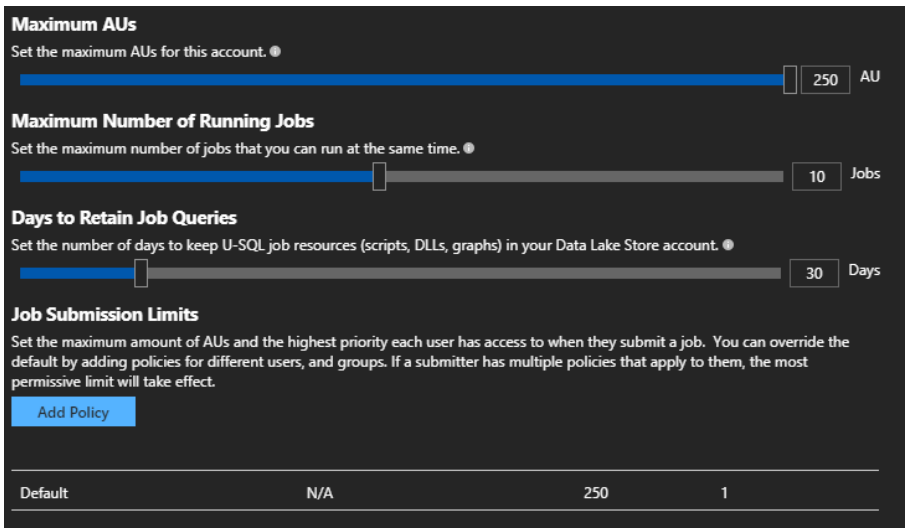
For the avoidance of doubt, to align with AW Security Policy, Encryption will be switched on in the ngBI Platform solution, and, this will be further recorded in the DLD produced for the ngBI platform.

4.2 Azure Data Lake Analytics (ADLA)

4.2.1 Monitoring

Monitoring is built in to the Azure Portal and available via PowerShell. This can provide an element of spend analysis. It would also be relatively straightforward to build alerts into the billing API when the ADLA spend has reached a certain amount.

There are limits that can be placed within each ADLA account controlling the number of compute units that can be requested by jobs, and the total AUs that can be running at any one time. The configuration for this is shown in the image below.



4.2.2 Performance and Scaling

U-SQL relies on parallelism to quickly process large amounts of data. It is possible to write jobs that are not optimised for parallelism, these jobs will still execute but will not use the allocated AUs efficiently. The team writing jobs therefore need to be trained to write effective queries and to use the in-built performance optimisers.

Jobs can be monitored for poor performance over time, however there is currently no provision for alerting around this, other than the limits described above.

4.2.3 High Availability & Disaster Recovery

ADLA jobs are deployed to the ADLA account and can be scheduled by any orchestration tool. When the job is executed, compute is provisioned. All metadata, logs and run-time files are stored within the Data Lake Store itself.

HA/DR is therefore not applicable for the bulk of the tool – the only element that could conceivably be lost within the ADLA account itself is the job history and logic. All U-SQL jobs should be developed within Visual Studio, and therefore added to source control, even if temporary/experimental jobs. These jobs can then be easily redeployed when required. As such, the tool is very highly resilient to fault/disaster.

4.2.4 Patches & Upgrades

The ADLA account is a Platform-as-a-Service component, any changes/patches to the service will automatically be rolled out to the ADLA service.

However, it is possible to deploy custom assemblies to the ADLA account in the form of custom libraries. These would need to be properly source-controlled and versioned as with any other application.

4.2.5 DevOps

The component itself can be deployed via an ARM template, however there is very little configuration required within the componet itself.

The Catalogue area of the Analytics Account, however, closely resembles a SQL Database. We would recommend the building of a CI pipeline within VSTS to deploy any changes to the catalogue automatically. There is no data held within the ADLA account itself, it is therefore a very lightweight deployment process.

4.2.6 Security

ADLA Security is based on Azure Active Directory. Users/Groups are added to the ADLA account and associated with a given role. They can also be added to a policy group, which governs the AU limits on the jobs they submit.

Data Access to the Lake inherits the access of the person submitting the job, which can be very finely controlled at the folder/file level.

4.3 Azure Analysis Services

4.3.1 Monitoring

As with other Azure components, AAS ties into Microsofts Operations Management Suite (OMS) which means that high level monitoring metrics can be collected in a central location. However, the data management views available within SQL Server Analysis Services are all also available – this means that any existing SSAS management queries can be repointed at the AAS service and used as normal.

There are also many tools available for evaluating the efficiency of deployed tabular models, such as the Vertipaq Analyser. This indicates where there are potential performance bottlenecks within the model itself, which is valuable when attempting to optimise available capacity and keep costs down.

4.3.2 Performance and Scaling

AAS supports both scale up and scale out approaches. Each model server can host several semantic models, the servers are sized as follows:

INSTANCE	QPUS	MEMORY (GB)	PRICE[1]
S0	40	10	£0.604/hour
S1	100	25	£1.513/hour
S2	200	50	£3.026/hour
S4	400	100	£6.045/hour
S8	320	200	£7.737/hour
S9	640	400	£15.48/hour

*prices are the generic price list, not the AWS EA rate.

The overall memory size is used to determine the total compressed size of models that can be hosted on this server, whereas the QPUs provide a measure of compute, concurrent queries and general processing speed. QPUs are formally defined as:

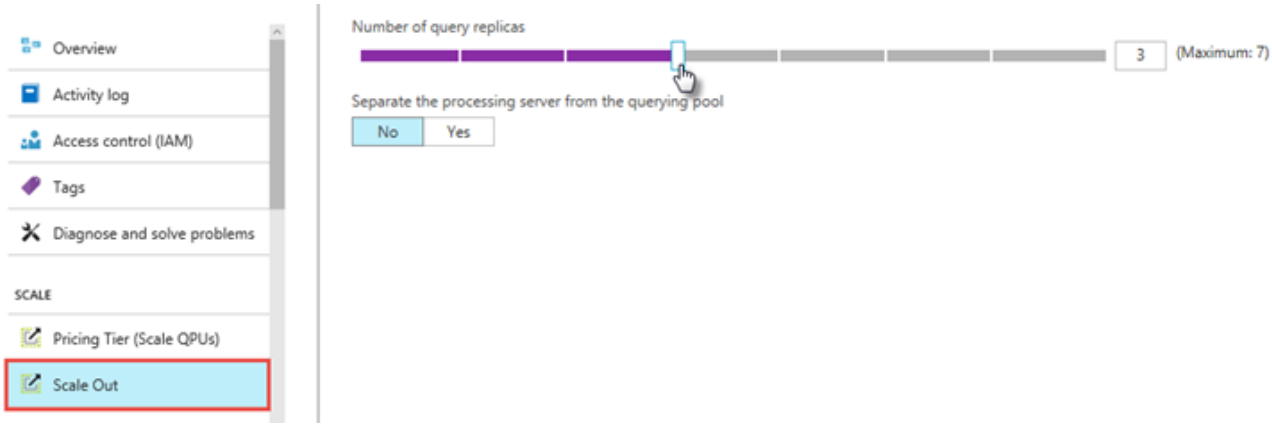
> *A Query Processing Unit (QPU) in Azure Analysis Services is a unit of measure for relative computational performance for query and data processing. A standard tier S4 instance (which offers 400 QPUs) provides four times the computational performance of a standard tier S1 instance (which offers 100 QPUs). As a rule of thumb, 1 virtual core approximates to roughly 25 QPUs, although the exact performance depends on the underlying hardware and the generation of hardware used.*

If a larger model size is used, you would need to scale up the AAS component to provide further memory. Whereas if more concurrency/query speed is required, you can instead scale out and host the same model with several read-only replicas.

For example, we currently have a server sized at S0 which provides 40 QPU and 10Gb costing £0.604/hour.

If we wanted to host a 15Gb model, we would scale this up to a S1 model, which provides 100 QPUs and £25Gb. This would now cost £1.513/hour.

However, if we were happy with the 10Gb size, but found that users were reporting slow query speed, we might want to scale it out instead. We would take our S0 server and create a *scale-out query replica*, which adds a 2nd S0 server to load balance query requests. Our 2 S0 servers now provide 80 QPUs at £1.208/hour – however we still have our 10Gb size limits.



We also have the ability to pause & resume the AAS components. If the model only needs to be available during office hours, for example, we can pause compute during evenings and weekends, we can easily save as much as 50% of the monthly server cost.

Pause/Resume and Scaling can all be controlled via the portal, or via PowerShell commands, which can be automated within Azure Functions or as Azure Automation jobs, neither of which add any significant cost to the architecture.

4.3.3 High Availability & Disaster Recovery

As with other PaaS components – the AAS component is highly-available and utilises failover components to provide a highly fault-tolerant service.

Disaster Recovery is also now available with Analysis Services. Previous implementations of Analysis Services Tabular models have assumed that any loss of data would require reprocessing from the source, which in our case would be the Data Lake Curated assets. This can be a timely process if the model contains a lot of data.

Azure Analysis Services processes data in a slightly different manner and can perform backup actions, writing the processed data down to a blob storage location. Restoring the model from this location is significantly faster than reprocessing from source as it no longer has to perform the columnar compression calculations.

This blob storage is managed by through the Azure subscription and so can be configured as geo-redundant to provide an additional level of disaster recovery.

4.3.4 Patches & Upgrades

The AAS component will be maintained with the most recent versions of the Analysis Services model automatically – it will receive new features as they are released, rather than waiting on the traditional SQL Server release & upgrade cycle.

4.3.5 DevOps

The DevOps story is very good around Azure Analysis Services. The service itself can be deployed via ARM templates or via individual PowerShell cmdlets.

Models itself can be incrementally deployed, managed and processed via PowerShell scripts which can be run locally or via Azure Automation or Azure Functions.

Also, the move from XMLA to TMSL (Tabular Model Scripting Language) has made code automation of Tabular models much more possible. Simple models can be generated from metadata and automatically deployed – it is rare to use this approach for complex, production models, but it can certainly be used to speed the development process and provide template models that developers can then extend with custom DAX calculations.

It is also possible to create template models and automatically provision filtered Data Marts, if there are certain models where the structure is common but users can only access certain subsets of the data model.

4.3.6 Security

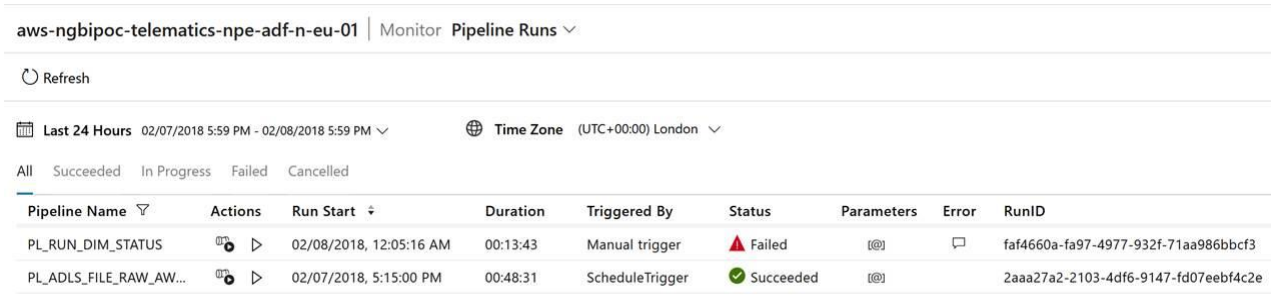
Azure Analysis Services is fully tied in to the Azure Active Directory security model. Users must have an AAD account in order to access the system, this is similar to traditional Analysis Services which relies on Windows Auth only.

Security can be managed on the model basis, providing users with access to one model or another. It can also be configured as row-level security – this ties access to a specific dimensional filter which has been mapped to their specific user id.

4.4 Azure Data Factory v2

4.4.1 Monitoring

Monitoring for Azure Data Factory (ADF) can be performed through the dedicated web UI "Author & Monitor" or programmatically. The web UI provides the ability to view the progress of Pipeline Runs, Integration Runtimes and Trigger Runs.



High level Information is displayed for each of these operations and can be drilled into further to show more detail, such as the included Activity Runs, used parameters or error messages. This information can be filtered by date or status.

4.4.2 Performance and Scaling

Azure Data Factory is a fully managed service with the capability to scale up and scale out. The following table, sourced by Microsoft, demonstrates how ADF performance depends on the allocated number of data movement units (DMU) and the throughput capacity of the target service.

			Cloud Sinks							On-prem Sinks**	
	(Unit: MBps)	# of cloud DMUs OR gateway nodes	Azure Blob (GRS)	Azure Data Lake Store	Azure SQL Data Warehouse (6000 DWU) PolyBase	Azure SQL Data Warehouse (6000 DWU) Bulk Insert	Azure SQL Database (P11)	Azure Table	Azure Cosmos DB (DocDB API, 2500RU) (single partition)	On-prem SQL Server	On-prem File System
Cloud Sources	Azure Blob (GRS)	4	56	56	1250	*	5	*	*	11	129
		8	**105**	**105**		*	9	0.2	*		
	Azure Data Lake Store	4	56	56	1060	*	5	*	*	10	114
		8	**120**	**108**		*	9	*	*		
	Azure SQL Data Warehouse (2000 DWU)	4	9	8	6	1	8	0.3	*	11	*
	Azure SQL Database (P11)	4	9	8	6	1	8	0.3	*	14	*
	Azure Table	4	2	2	*	2	2	1	*	1	*
	Azure Cosmos DB (DocDB API, single partition)	4	2	2	*	2	2	*	*	*	*
	Amazon S3	8	**107**	**101**	69	*	*	*	*	*	*
	Amazon Redshift	4	*	*	7.2	*	*	*	*	*	*
On-prem Sources	On-premises SQL Server	1	7	7	18	0.4	7	0.2	0.04	*	*
	On-premises File System	1	195	192	102	0.3	6	0.2	0.04	*	*
		4	**505**	**510**	*					*	*
	On-premise HDFS	1	179	183	83	0.3	3	0.2	0.04	*	*
		4	**500**	**525**	*					*	*

Unit: MBps

**: The throughput numbers for this source-sink combination will be published later.*

***: For copying from cloud sources to on-prem sinks, single Self-hosted Integration Runtime node was used.*

When moving data with Azure Integration Runtime in ADFv2, the minimal allowed DMUs is 2. Better performances can be achieved on a cloud-to-cloud copy activity run by either scaling up ADF with more DMUs (e.g. increasing ADF to 100DMUs can achieve a throughput of 1.0GBps when copying data from Azure Blob into Azure Data Lake) or scaling up and scaling down the target service as needed (e.g. scale up Azure SQL DB to 50 DTUs during data transfer and scale down once the activity is completed).

It is important to remember that, with ADF, we should always aim for optimal performance, since we are charged based on the total time of the copy operation. The total duration billed for data movement is the sum of duration across DMUs.

4.4.3 High Availability & Disaster Recovery

High Availability is secured in ADF via the Data Management Gateway, which can have up to 4 nodes. This means that, if a node goes down for some reason or needs to be taken offline for maintenance, the other nodes will still be available for moving the data.

Because ADF does not store any data, disaster recovery is not applicable.

4.4.4 Patches & Upgrades

The ADF account is a Platform-as-a-Service component, any changes/patches to the service will automatically be rolled out to the ADF service.

4.4.5 DevOps

At the time of writing there isn't a VSTS project that we could use for CI implementation or ARM template for ADF v2, therefore, the alternative is the use of custom PowerShell deployment scripts.

4.4.6 Security

Azure Data Factory itself does not store any data except for linked service credentials for cloud data stores, which are encrypted using certificates. As an alternative, the data store's credentials can be secured in Azure Key Vault.

4.5 Azure SQL DB

4.5.1 Monitoring

There are a number of different ways to monitor an Azure SQL Database.

The portal blade gives the ability to see high level details for the database like the storage size and usage, it also provides the ability to create alerts based on configured rules and conditions. Diagnostics can also be enabled from here which can provide a time log of further information.

Azure SQL DB can also be connected to from SQL Server Management Studio and be monitored using the existing traditional methods.

4.5.2 Performance and Scaling

Azure SQL DB is scaled by adjusting the DTUs (Database Transaction Units) which it has assigned, a DTU is a combined measure of CPU, memory, I/O (data and transaction log I/O). The amount of DTUs available as well as the database storage size is based on the service tier which the Azure SQL DB is assigned to.

	Basic	Standard	Premium	Premium RS
Maximum storage size*	2 GB	1 TB	4 TB	1 TB
Maximum DTUs	5	3000	4000	1000

The amount of DTUs assigned to an Azure SQL DB is an indication of the relative amount of resources between Azure SQL DBs at different performance levels and service tiers, for example a Standard database with 50 DTUs will have 10 times more DTU compute power than a Basic database with 5 DTUs.

Changing the Service Tier or amount of assigned DTUs for a database will create a replica of the existing database at the new performance level and switch connections over to this new replica. Although no data is lost during this process the database will be temporarily unavailable, the length of time for this operation varies depending on a number of factors. This should be kept in mind when planning an upgrade or downgrade.

4.5.3 High Availability & Disaster Recovery

Azure SQL DB is of a PaaS architecture and High Availability is built in to the service, therefore no further configuration or maintenance is required. Microsoft applies an SLA and maintains full control over the High Availability system configuration and operation.

The SLA for High Availability is offered by Microsoft at a region level and does not cover total region failure due to factors such as natural disaster, acts of terrorism etc.

For any type of storage used by Azure SQL DB, Microsoft provides the following key benefits:

- Customers get the full benefit of replicated databases without having to configure or maintain complicated hardware, software, operating system, or virtualization environments.
- Full ACID properties of relational databases are maintained by the system.
- Failovers are fully automated without loss of any committed data.
- Routing of connections to the primary replica is dynamically managed by the service with no application logic required.
- The high level of automated redundancy is provided at no extra charge.

Azure SQL DB uses a failure detection system based on the Azure Service Fabric. All reads and writes take place on a primary replica, when a failure is detected on this primary replica a secondary replica is promoted to the status of primary.

4.5.4 Patches & Upgrades

Much like the other PaaS components Azure SQL DB is automatically updated and patched without any need for manual intervention.

4.5.5 DevOps

The database source code is stored in Source Control and can easily be packaged and deployed to any required environment. The code is also environment redundant so can be deployed to Development, UAT, Pre-Production and Production environments without any further configuration changes.

4.5.6 Security

Security is applied to Azure SQL DB with all of the expected features of the latest SQL Server version including logins, permissions, roles etc. There are also security features and resources applied as standard these are:

- Encryption – Transport Layer Security for data in motion, Transparent Data Encryption for data at rest and Always Encrypted for data in use
- Access Control – Firewall and firewall rules, authentication with both SQL and Azure Active Directory Authentication, Authorization and Row-level security
- Proactive Monitoring – Auditing and Threat detection
- Compliance – Regular audits and certified against a number of compliance standards

4.6 Azure SQL Data Warehouse

4.6.1 Monitoring

SQL DataWarehouse can be monitored by many existing SQL Server monitoring tools, however we are intending to use is as a lightweight serving layer.

Adatis have produced several reporting queries for monitoring Polybase usage and overall concurrency consumption, these could be easily tied into a central monitoring solution if required.

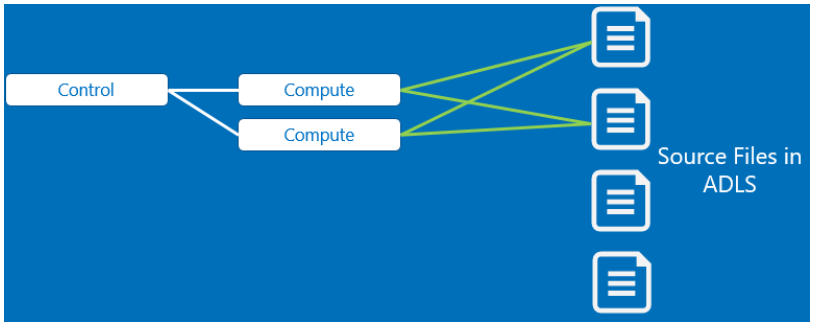
As with other Azure components, there are several high-level metrics available via the Azure portal and, therefore, through OMS.

4.6.2 Performance and Scaling

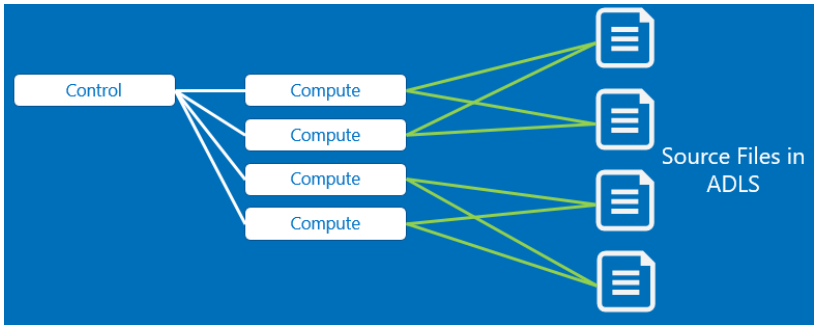
SQL Datawarehouse scales in the form of Data Warehouse Units (DWUs), each of 100 DWUs equates to an additional compute server added to the MPP cluster.

Each compute node provides additional querying threads against the internal storage distributions, but also external PolyBase querying threads. Therefore, by increasing the number of DWUs allocated to the server, we can directly improve flat file querying speeds.

For example, if we have four very large files to query and we were scaled at 200 DWUs, we would have 16 external threads to access sections of these flat files:



If we scale the SQLDW up to 400 DWUs, we now have 36 thread that can be reading external files simultaneously:



This scaling can be performed using the Azure Portal, PowerShell cmdlets or T-SQL within the parent logical SQL Server.

4.6.3 High Availability & Disaster Recovery

SQL DataWarehouse uses premium Azure storage blobs as the primary underlying storage. This storage is naturally redundant (every blob is copied on to three disks and will transparently failover & rebuild if hardware issues occur). The storage layer is, therefore, highly available.

One of the fundamental tenants of SQLDW is the separation of storage and compute – the compute is provided by generic DW server layers and can be switched out at any time. We therefore have a similar failover model for compute whereby if any one compute node encounters a failure, a new one will be swapped in. This will impact any currently-running transactions but will otherwise be a transparent failover.

Disaster Recovery with SQLDW is controlled via automatic snapshot backups taken every 7 hours. These backups are geo-redundant and can be restored to a paired geographic region very easily. These rolling backups are maintained for 7 days. Due to the PaaS nature of SQLDW, there is little configuration that can be done around these backup timings.

In our case, we are not expecting to use SQLDW for data storage, only as a lightweight querying layer – we therefore only have to store the schema metadata which can be redeployed to a completely new SQLDW instance with minimal downtime.

4.6.4 Patches & Upgrades

As a PaaS component, the SQLDW will automatically receive updates and patches throughout its lifetime. In the case of critical issues, patched servers will be hot-swapped into the current compute allocation to bring it up to the latest version without any downtime.

4.6.5 DevOps

DevOps is a usually a pain point with SQLDW currently as the traditional methods of performing automated deployments for database projects are not yet available for SQLDW – this means that managing incremental change within SQLDW is a painful process.

However, as we are not storing data within the SQLDW layer, we can use custom PowerShell deployment scripts to push our external table metadata out each time with minimal impact to the existing table – you would normally avoid dropping & recreating table objects to avoid losing data, but as we are not storing data this approach is sufficient.

4.6.6 Security

User security within Azure SQLDW is handled much like the Azure SQLDB offering – you have the full SQL Server functionality of roles, entity permissions etc. SQLDW is compatible with Azure Active Directory and so we would recommend that AD Groups are designed and allocated to custom database roles which, in turn, provide the relevant grant permissions on the objects and schemas required. The current usage of SQLDW would indicate that only two roles are needed – general admin and a read-only user group.

4.7 Power BI

4.7.1 Monitoring

Currently auditing and monitoring of Power BI is split between three locations:

- Inside the Power BI Service (PowerBI.com)
- Inside the Office 365 Administration portal
- Inside report viewer (Usage metrics)

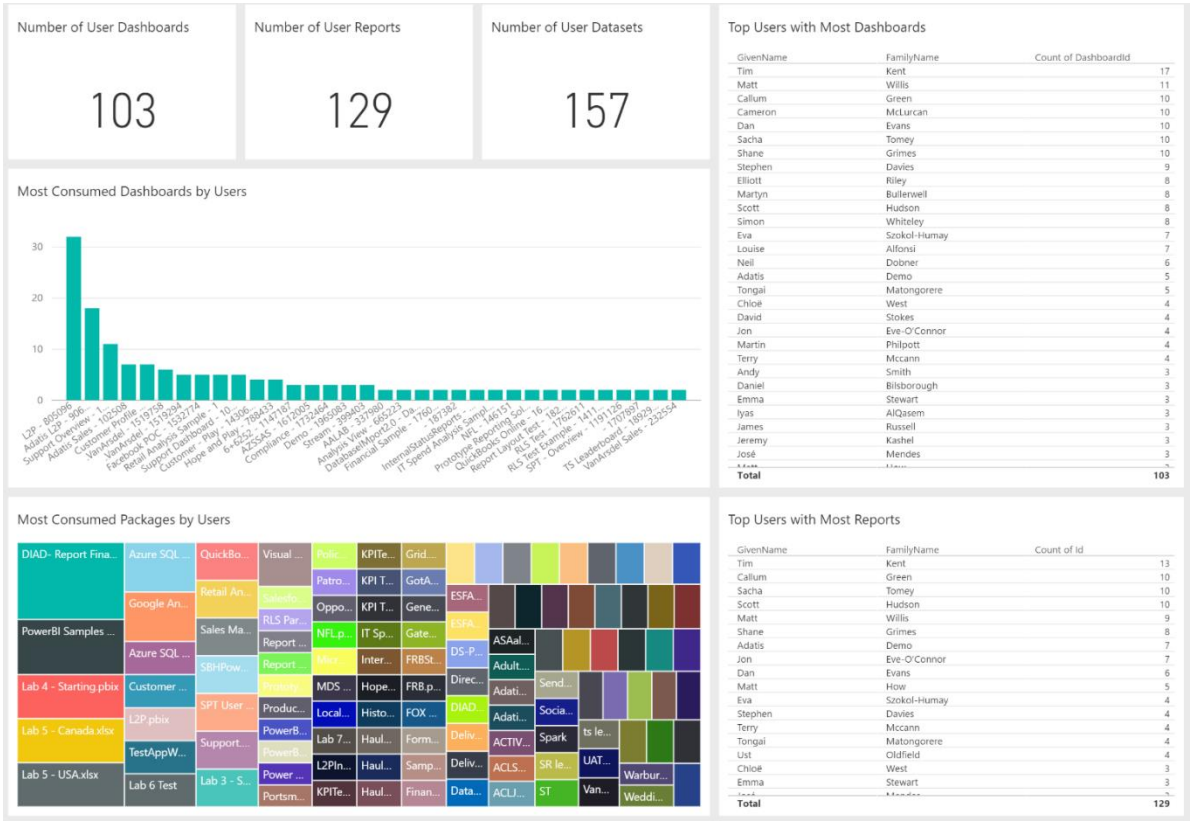
4.7.1.1 Power BI Service usage metrics

The metrics available inside the Power BI service are a high-level overview of Power Bi usage across the entire tenancy. The available metrics can be seen below:

- Total counts of Dashboards, Reports and Datasets.

- High level statistics around most used reports/dashboards
- High level statistics around user interaction and user usage

An example of the usage metrics can be seen below.



4.7.1.2 Audit logs in Office 365 administration portal

The Audit logs provide a more substantial level of detail to Power BI activities, below is a list of activities that can be added to the audited:

- Viewed Report/Dashboard
- Created Report/Dashboard
- Edited Report/Dashboard
- Deleted Report/Dashboard
- Shared Report/Dashboard
- Printed Report/Dashboard
- Exported Report/Dashboard
- Downloaded Report
- Published report to Web
- Deleted Datasets
- Create Groups
- Added a Group Member
- Create Content Pack
- Updated Organizations Power BI settings

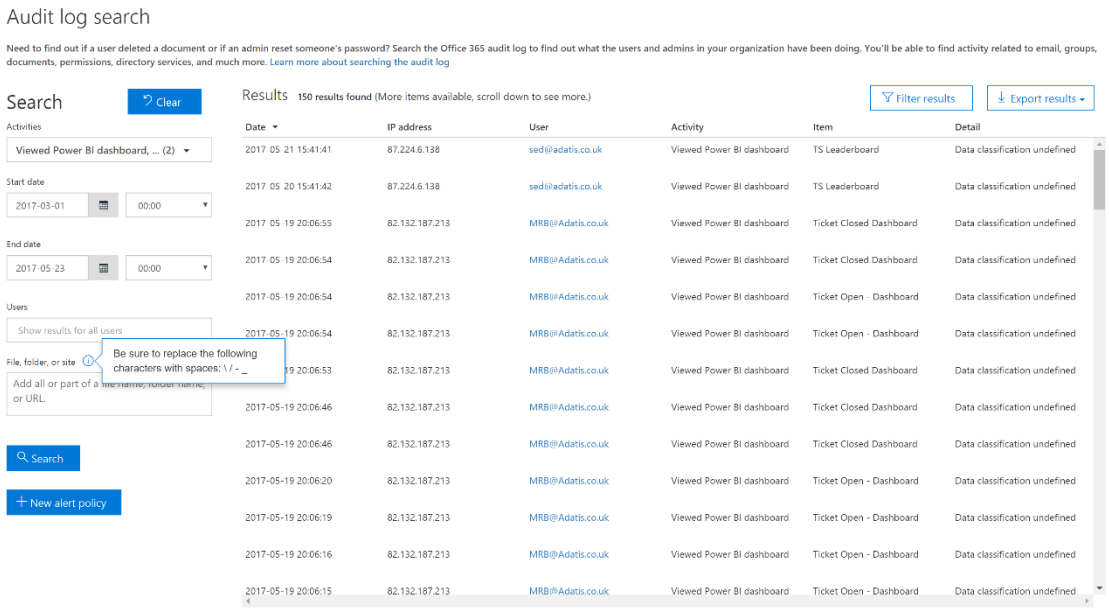
- Analysed Dataset

The logs in O365 are held for 90 days, and there can be (up to) a 24-hour delay before being shown.

The logs can be searched through the O365 Admin Portal, filters can be placed to help user navigation around the data.

These logs are only available to Power BI Global Admins.

An example of the O365 audit logs can be seen below:



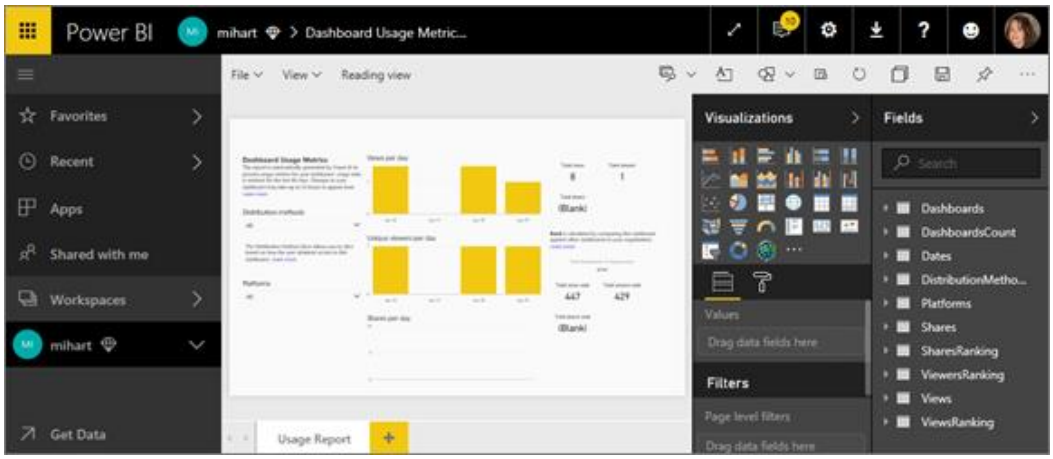
The results from a user query can be exported directly from the O365 Admin Portal for further analysis. Data can also be directly exported using PowerShell, please see below for details.

https://powerbi.microsoft.com/en-us/documentation/powerbi-admin-auditing/

4.7.1.3 Report usage metrics

Report Usage metrics, a new feature to Power BI, are report usage statistics available directly to the report builder and admin. This makes them unique compared to other report statistics in Power BI that are only available to Power BI Global Admins.

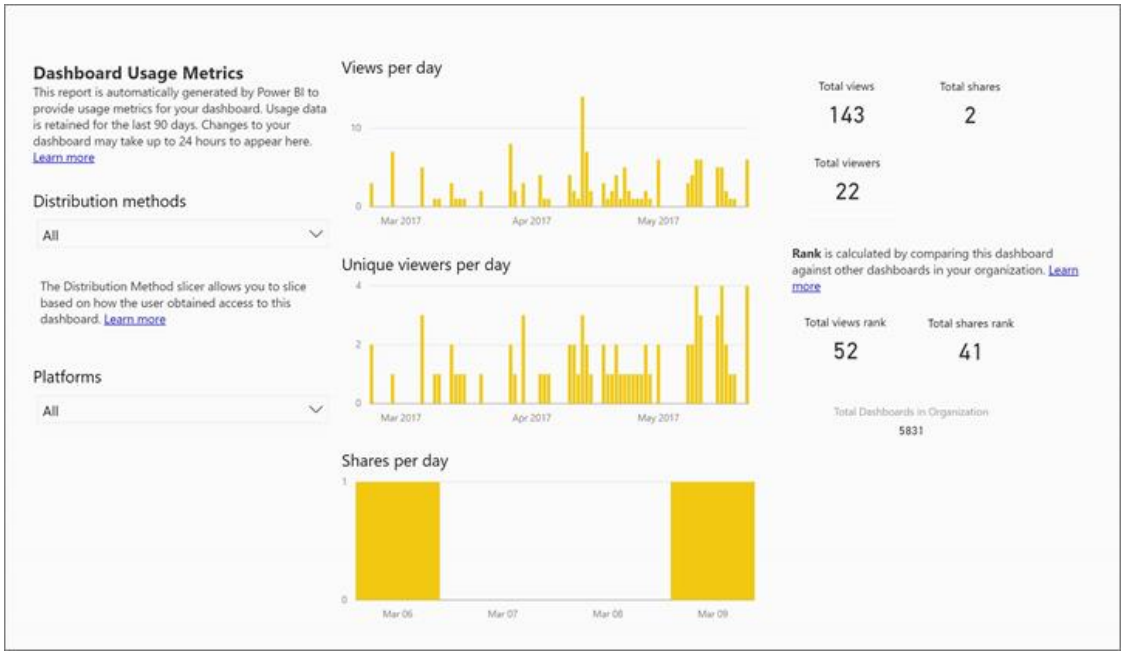
These usage metrics reports are read-only. However, you can personalise a usage metrics report by using "Save as." This creates a brand-new dataset and converts the read-only report to a full-featured Power BI report that you can edit. Not only does the personalised report contain metrics for the selected dashboard or report, but by removing the default filter, you have access to usage metrics for all dashboards or all reports in the selected workspace. An example of this can be seen below:



Report usage metrics display provides the following functionality:

- Displays metrics for the last 90 days
- Slice based on platform: Web, Mobile
- Slice based on Distribution Method: Direct, Share, Content Pack, Power BI Apps
- Works for both Reports and dashboards

An image of Report usage metrics can be seen below:

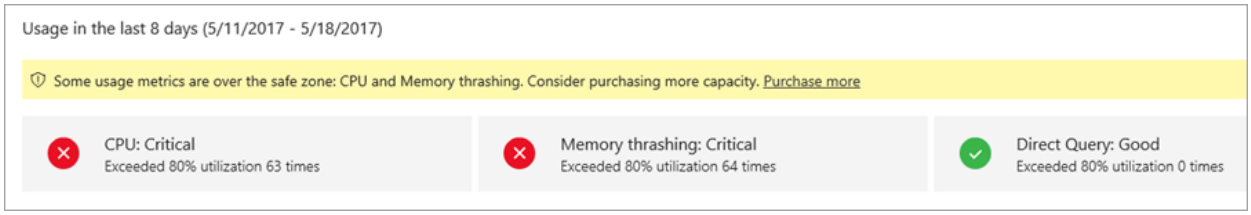


4.7.1.4 Auditing Power BI Premium

Power BI Premium introduces its own auditing features which can be accessed from inside the Power BI Portal. Each capacity in Premium is assigned a Capacity Admin/s, these admins are who has access to these usage metrics.

For each capacity in Power BI Premium, you will be able to use usage measurements for CPU, memory and Direct Query. Each KPI has three indications, Good (green), Marginal (yellow) and Critical (red).

Monitoring of the Premium Capacity is essential to drive the scale up or scale down of power, therefore giving the right balance between cost and performance to the user.



Metric	Description
CPU	CPU usage of your cores.
Memory	Represents the memory pressure of your backend cores. Specifically, this is a metric of how often models are evicted from memory due to memory pressure from usage of multiple models.
DQ/s	* We limit the total number of DirectQuery and live connection queries per second. * The limits are 30/s for P1, 60/s for P2 and 120/s for P3. * DirectQuery and live connection queries count equally to the above throttle. For example, if you have 15 Direct Queries and 15 live connections in a second, you hit your throttle. * This applies equally to on-premises and cloud connections.

There is a limitation on the number of Direct Query connections per second that can be made, the limit is dependent on the size of the premium capacity (P1:30, P2:60, P3:120).

When these metrics are marginal/critical, your users may see degradation of report and refresh performance, especially during peak load times.

Metrics reflect utilisation over the past week, and are designed to count instances when the capacity is overloaded, and is therefore providing less-than-optimal performance for your users.

Each occurrence of utilisation over 80% should be considered a potential case of performance degradation. Too many cases are a good indicator of significant performance problems for users.

4.7.2 Performance and Scaling

When using the standard (not Premium) Power BI license model there is no concept of scaling capacity, and therefore performance needs to be controlled via the performance of the source system.

In the recommended Enterprise Usage model for Power BI, reports utilise Direct Query mode, whereby all processing of data is done on the source system, and in the case of this solution Analysis Services. This reduces the impact on the Power BI service, and therefore supporting more concurrent users without degradation of performance.

If using the Power BI Premium licensing model then performance can be controlled via the amount of capacity purchased. More capacity gives more memory and processing power to your capacity.

4.7.3 High Availability & Disaster Recovery

4.7.3.1 High Availability

Power BI is Software as a Service (SaaS) which means it is entirely managed by Microsoft.

The Service Level Agreement (SLA) from Microsoft regarding the uptime of Power BI can be seen below:

"We strive to keep the Service up and running; however, all online services suffer occasional disruptions and outages, and Microsoft isn't liable for any disruption or loss you may suffer as a result. You should regularly backup the Customer Data that you store on the Service. Having a regular backup plan and following it can help you prevent loss of your Customer Data."

Despite the relaxed nature of the SLA for Power BI, in Adatis' experience the uptime of Power BI has been very good.

Since Power BI's release in July 2015 we have only experienced one intermittent partial outage. The UK data centre had limited connectivity for around 6 hours, this was in the early months of Power BI's release and no data or reports were lost, only limited service for the time. Since this time, we have not experienced any disruptions to the service.

Further details around the Power BI SLA can be found here: https://powerbi.microsoft.com/en-us/terms-of-service/

4.7.3.2 Disaster Recovery

It is recommended that all Power BI reports are built using Power BI Desktop and then deployed to the Power BI Service. All Power BI Desktop files can then be appropriately source controlled and managed for future deployments and disaster recovery.

4.7.4 Patches & Upgrades

The Power BI Service is a Software as a Service (SaaS) platform. Updates to the service are released autonomously and from a user and organizational perspective there is no control over this. This is a common approach for SaaS platforms.

Significant regression testing is undertaken by Microsoft when rolling out updates to ensure that it is backwards compatible with existing Power BI assets.

Microsoft provide a monitoring dashboard that can be used to check service levels in your region. If an issue does arise based on changes made via a Microsoft update, then a support ticket should be raised with Microsoft.

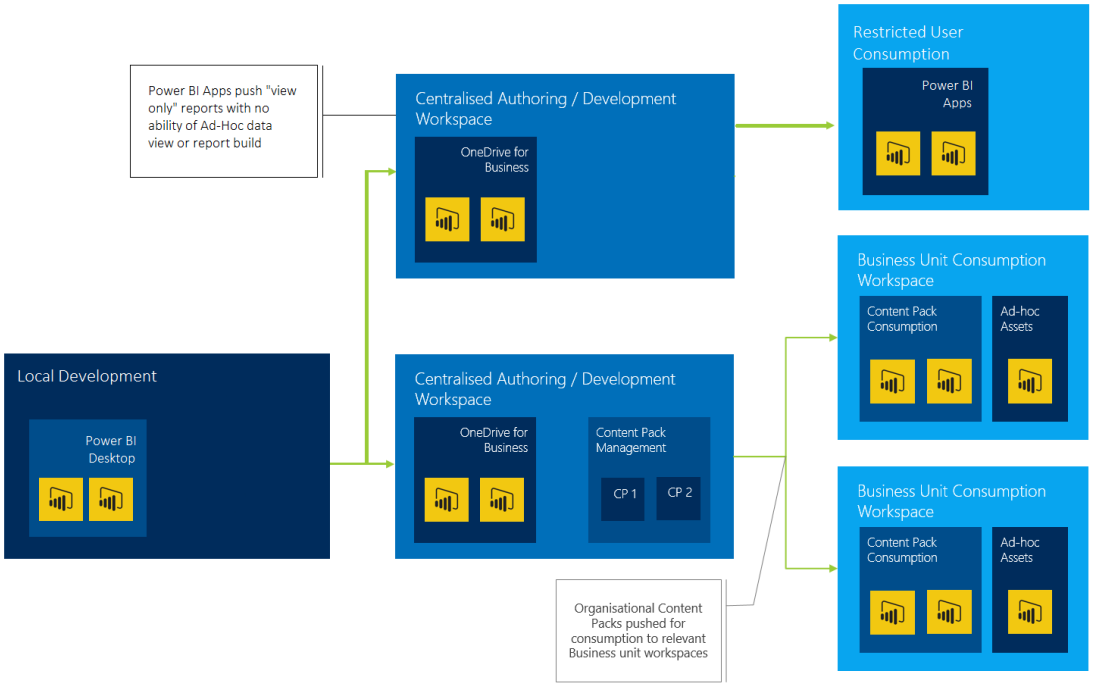
Support for **custom visualisations** in Power BI are not covered by Microsoft's regression testing processes as the custom components are often developed by third-parties.

4.7.5 DevOps

There are myriad mechanisms for storing and organising the assets that comprise a Power BI solution. Due to the nature of AWS, the following high-level approach provides a consistent mechanism that (a) provides control to the BI Team, (b) manageability from an overriding IT / BI Team perspective, and (c) flexibility, ownership and accountability to each of the Business Units requiring access to information and analytics.

The following diagram articulates the structure of the storage and control mechanism and which relies on liberal use of Power BI Workspaces and Power BI Apps.

When a Power BI Workspace is created, an Office 365 (cloud only) Group is created – this provides a membership administration container and OneDrive for Business file share area that benefit from features available within SharePoint online, such as versioning, workflow and check-in/check-out.



Essentially, the development model is based on a hub and spoke arrangement. Which is the recommended approach to ensure management practices align as best as practically possible to traditional development practices. The approach addresses many of the inherent shortcomings associated with Power BI.

The Hub and Spoke arrangement is the recommended pattern for the deployment and enterprise level organization of Power BI content.

The above diagram describes two hubs with many spokes but it is possible to introduce more hubs, perhaps separated by subject areas or subject matter expertise within the BI team. Or, hubs could in turn be owned and managed by business units themselves, publishing information to sub-business units.

Separate hubs are required for Power BI Apps and for organizational content packs.

There is a one to one relationship between Power BI Workspaces and Power BI Applications. This means that all assets within the development workspace are included in the application. Contrary to this, content packs are more selective and can contain a subset of assets contained within a development workspace enabling a one to many relationship between development workspaces and content packs and thus Business Unit Workspaces (spoke).

For clarity, two hubs with two spokes arrangement using Content Packs and Apps is the focus of this document. Additional commentary will be applied where the process or concepts differ for Power BI Apps.

4.7.5.1 Local development

It is recommended to create all assets using Power BI Desktop and avoid creating assets in the Cloud service directly. The exception to this recommendation are Dashboards. Dashboards can only be created through the Cloud service.

It is recommended to create all assets using Power BI Desktop and to avoid creating assets in the Cloud service directly

.

Once reports have been created in Power BI Desktop, the .pbix files should be uploaded to the SharePoint location associated with the 'Development' Hub Workspace to provide version control and centralised file management.

4.7.5.2 Centralised Authoring / Development Workspace

These workspace act as the hubs and should be owned and managed by the BI Team. It is a communal Development workspace from which to test and publish content that has already been developed in Power BI Desktop.

Following approval, content can be grouped together and placed in a Content Pack for publication and shared with Business Unit Workspaces for consumption.

In the case of Power BI Apps, the same principal applies with the constraint that a Centralised Authoring / Development workspace will be required for each application.

4.7.5.3 Business Unit Consumption Workspace

The workspaces that serve as the spokes provide a communal consumption area for each Business Unit or collective audience. The purpose is to consume centrally managed Power BI assets (Created in the Development Workspace above) and for Business Unit members to work on ad-hoc requirements in isolation. These ad-hoc requirements may, or may not be based on centrally managed assets.

4.7.5.4 Restricted User Consumption

For scenarios where a prescribed, read-only (but still interactive) view of content is required, the recommended approach is to utilise Power BI Apps. Apps are created from content within a development workspace (Hub).

Development workspaces have a one to one relationship with Power BI Apps. This means that all assets within the development workspace are included in the application. Contrary to this, content packs are more selective and can contain a subset of assets contained within a development workspace enabling a one to many relationship between development workspaces and content packs and thus Business Unit Workspaces (spoke).

Users or groups with access to Apps will have full read and interactive capabilities provided by the reports and dashboards but they will not have the ability to modify the reports, manipulate or view the data set, or create new reports based on the data set.

4.7.6 Security

Security in Power BI is applied via Azure Active Directory users and groups.

Security can be applied and the following levels

- Workspace
- Power BI App
- Individual report level

When users are creating new datasets, they will also require access to the Analysis Services model to obtain data. This is given via Azure Active Directory Users or Groups.

The credentials used for accessing the data source are based on the credentials entered when the data set was set up, not the credentials of the user creating the report. Therefore, it is recommended that Service Accounts are setup and used for the creation of datasets inside Power BI, this will result in all users accessing the source data via the service account and not the user that initially set up the data set.

4.7.7 Self Service Approach

Historically a very centralised approach was taken to BI, whereby content was created only by the BI team and pushed out to users.

Presently, we are seeing the Self-Service approach taken often, where users have more control over BI elements and can create their own reports and potentially data sets.

For self-service BI to be implemented successfully, it requires changes to the BI process that fosters a two-way relationship between users and the BI team. This contradicts the way one push of content we have previously seen.

As users embrace the self-service approach, questions around BI tools are inevitable, to counter this, users should have an opportunity to speak to a designated person (or people) within the BI team to answer questions, provide tutorials and be a point of contact with queries around BI.

The second issue that can arise from self-service is the retrospective creation of measures back into the original dataset. As users create their own measures, it's easy to end up with duplicates of the same measures being created. Commonly used measures, used by multiple business units, should be transitioned to the central semantic layer and readily available to users rather than needed to be created by multiple users themselves.

Both these points should be handled by the same point of contact within the BI team. They are the face of the BI team to users and orchestrates effective and efficient self-service of users, as well as feedback to the BI team items for development and new user requirements.

Below are some responsibilities that this role should entail:

- Being a custodian of BI strategy within the organization, ensuring that the BI strategy is shared, understood, and implemented

- A focused business function to ensure the adoption and use of BI best practice for both tools and techniques
- The leveraging of existing BI solutions. This relates not only to the tools and data, but also the intellectual property (IP) that resides in both the business and BI areas
- The coordination, consolidation, and prioritisation of BI initiatives
- The supporting of the business users in defining and articulating their requirements
- The provision of more agility in reacting to business changes
- Reducing risk when implementing new BI projects/solutions
- Cost reduction and savings by consolidating effort and removing the need for 'local' or 'desktop' BI solutions
- The provision of a forum for knowledge sharing and knowledge re-use
- The provision of a forum for collaboration

An Agile methodology to BI implementation supports faster turnaround time for functionality and benefit to users.

A good implementation of an Agile way of working, includes the ability for short release cycles.

We believe this can be achieved by the Agile methodology and the use modern technology.

4.8 Data Science Virtual Machine

4.8.1 Performance and Scaling

Performance of the Data Science VM can easily be controlled via scaling the VM memory and power up/down.

To save on costs it is recommended that the VM is scaled down or turned off when not being used and that an appropropriate amount of power is assigned at any given time.

Scaling of the VM can be automated and controlled via a PowerShell script and/or Azure Automation. This will allow the scheduling of shutting down the VM out of business hours or gives the Data Science team the power to execute the script to scale up as and when needed without needing to raise support calls.

4.8.2 High Availability & Disaster Recovery

Due to the nature of this VM High Availability and Disaster Recovery have lesser importance in comparison to other components in the solution.

Data Science models created on this box should always be stored in a version control system such as GIT or TFS. All data sets utilised by models on this box should remain in the Data Lake and therefore no need for HA/DR.

4.8.3 Security

The VM should be added to the AWS Azure domain, this will enable the security and access to the VM to be controlled via Azure Active Directory.

4.9 Azure Machine Learning Model Management Services (AML MMS)

4.9.1 Performance and Scaling

Azure Machine Learning Model Management Services (AML MMS) has fantastic built in "Auto-Scale" functionality. These settings automatically increase the number of deployed containers based on the load within the existing cluster. They also control the throughput and consistency of prediction latency.

Scale settings

Scale Type	Auto
Number of replicas	1 replicas
Min replicas	1 replicas
Max replicas	10 replicas
Target utilization	70 %
Refresh period	1 seconds

What this results in is the ability to set Minimum and Maximum amounts of capacity, along with an ideal threshold of what we aim to have the cluster running at, and then the service is able to scale up and down as needed to maintain the required level of service.

4.9.2 High Availability & Disaster Recovery

The cluster/container style architecture of AML MMS is based on the Kubernetes architecture which natively supports high availability as work is distributed and load balanced between many different clusters of nodes.

4.10 Azure Data Catalog

The Azure Data Catalog is a fully managed cloud service that provides any user (analyst, data scientist or data developer) with the capability to discover, understand and consume data sources available in the organization. The crowdsourcing model of metadata and annotations provides the organization with the capability to 1) enrich the data assets registered in the catalog at any time; 2) let users easily understand the purpose of the data and how it is being used within the business; 3) allow the IT department to maintain the control and oversight over all the data sources.

Azure Data Catalog provides all users with the capability to discover and register a range of enterprise data sources (list of supported data sources available here). To register the data sources, Azure Data Catalog offers two options, 1) manually add the data via the Azure Data Catalog portal; 2) use the data source registration tool.

Once the data source is registered, the registration tool extracts information about the structure of the selected objects. The information, also known as structural metadata, contains the object's location, name and type. For the supported data sources, the tool can also automatically extract the attribute/column name and data type. Finally, for the table and view based data sources, the tool supports the inclusion of a snapshot preview of the data up to 20 records.

Azure Data Catalog also has a REST API that provides programmatic access to the catalag resources to register, annotate and search data assets. It is possible to use a custom application and send HTTP requests with a GET, POST, PUT or DELETE method to an endpoint that targets a resource in the catalog.

Azure Data Catalog uses two primary mechanisms to discover data sources, searching and filtering. Searching is one of the most important mechanisms in the service, providing the user with the tools to run basic and

complex searches. When searching for a specific asset, the service will return everything that matches against any property in the catalog (data source type, project type, tags, experts…), including comments and annotations. Filtering is mainly designed to complement the searching and constrain the search results to matching assets.

Following is a list of functionalities tested in this solution:

Functionality	Description
Registration of multiple data sources	The following data sources were added to the catalog: • Azure Data Lake files (RAW/BASE/ENRICHED/CURATED) • Azure SQL Data Warehouse • Azure Analysis Services • Power BI

Definition of a Business Glossary	• Ability to create a business glossary for governed tagging (only available in the Standard Edition)
	• Each glossary term contains:
	  ○ a business definition
	  ○ a description to capture the intended use for the asset
	  ○ a list of stakeholders (only users available in the Azure Active Directory can be added)
	  ○ a parent term, which defines the hierarchy in which the term is organized

Documentation and annotation capabilities	• Ability to document data sets using a rich text editor with paragraph formatting (e.g. headings, text formatting, bullet and number lists and tables) • Capacity to reference content from another content repositories, such as SharePoint or a file share • Support of different annotation types (e.g. friendly names, description or user/glossary tags)
Capability to work with big data sources	Big data sources are usually organised in directories and subfolders. To avoid cluttering the catalog, Azure Data Catalog allows the registration of directories or individual files.
Discovering capabilities	• Ability to use free text to search for specific elements • Capacity to search using comparison and Boolean operators • Ability to reuse saved searches • Data lineage and data profiling capabilities
GDPR Compliance	• Identification and classification of personal data according to the GDPR terminology • Categorization and labelling of the data sources • Implementation of security policies at the service and data source level • Definition of ownership per data source

4.10.1 Monitoring

The Azure Data Catalog monitoring is available via the Azure Portal and the Azure Data Catalog Portal. The Azure Portal provides information about the Pricing tier and can be used to restrict access to the service via the Azure Portal. The Azure Data Catalog Portal provides details about the number of registered assets and users. These metrics are can be used to calculate the cost of the service per month.

4.10.2 Performance and Scaling

The Standard Edition of the Data Catalog supports up to 100,000 registered data sets and unlimited number of users. The limitations of the REST API can be found here.

4.10.3 High Availability & Disaster Recovery

Azure Data Catalog supports geo-replication for business continuity and disaster recovery. All contents, including data source metadata and crowdsourced annotations, are replicated between two Azure regions at no additional cost to customers (further details here).

4.10.4 Patches & Upgrades

Azure Data Catalog is a Platform as a Service (PaaS) platform. Updates to the service are released autonomously and from a user and organizational perspective there is no control over this.

4.10.5 DevOps

This section is not applicable to the Azure Data Catalog.

4.10.6 Security

Azure Data Catalog Security is based on Azure Active Directory. Users/Groups can be added in the Azure Data Catalog Portal as Catalog users and/or Glossary Administrator users. They can be added to each data source and granted with Read/Write, Read or No Access privileges. Catalog Users can register, annotate, discover and use data sources according to the privileges added to the user or security group. To run the Data Catalog registration tool, the user needs permissions on the data source that allows him to read the metadata from the source.

5 Project Delivery

5.1 Implementation Approach

Development teams should follow an iterative, agile approach to developing each application whilst working under the umbrella of Anglian's programme management framework. Scrum is one such agile approach to building solutions and software and is briefly described in the following sections:

5.1.1 Scrum

Scrum is the leading Agile product development framework. The concept that would become Scrum was first introduced to the world in 1986 by Hirotaka Takeuchi and Ikujiro Nonaka in the "New New Product Development Game" (Harvard Business Review, January/February 1986). They defined their approach as a "flexible, holistic product development strategy" and proposed it would result in fast, flexible product development. They called it the holistic or "rugby" approach because, much like in a rugby match, one cross-functional team passes the "ball" back and forth on the way to the "goal line." This was, and continues to be, in stark contrast to approaches that progress in a rigid, linear fashion.

5.1.2 High Level Scrum Process

Scrum is a team-based approach to delivering value to the business. Team members work together to achieve a shared business goal. The Scrum framework promotes effective interaction between team members so the team delivers value to the business.

Once the team gets a business goal, it:

- Figures out how to do the work
- Does the work
- Identifies what's getting in its way
- Takes responsibility to resolve all the difficulties within its scope
- Works with other parts of the organization to resolve concerns outside their control
- This focus on team responsibility in Scrum is critical.

Scrum requires a working, finished product increment as the primary result of every sprint. Whatever activities take place during the sprint, the focus is on the creation of the product increment. A Scrum team's goal is to produce a product increment every sprint. The increment may not yet include enough functionality for the business to decide to ship it, but the team's job is to ensure the functionality present is of shippable quality.

Scrum is a framework designed to promote and facilitate collaboration. Team members collaborate with each other to find the best way to build and deliver the software, or other deliverables, to the business. The team,

especially the product owner, collaborates with stakeholders to inspect and adapt the product vision so the product will be as valuable as possible.

Scrum teams make frequent plans. For starters, they plan the current sprint. In addition, many teams create longer-term plans, such as release plans and product roadmaps. These plans help the team and the business make decisions. However, the team's goal is not to blindly follow the plan; the goal is to create value and embrace change. In essence, the thought process and ideas necessary for planning are more important than the plan itself.

A plan created early is based on less information than will be available in the future so, naturally, it may not be the best plan. As new information is discovered, the team updates the product backlog. That means the direction of the product likely shifts. This continuous planning improves the team's chances of success as it incorporates new knowledge into the experience.

Scrum teams constantly respond to change so that the best possible outcome can be achieved. Scrum can be described as a framework of feedback loops, allowing the team to constantly inspect and adapt so the product delivers maximum value.

Scrum Roles & Responsibilities

A Scrum team has three roles:

- • Product Owner - holds the vision for the product
- • ScrumMaster - helps the team best use Scrum to build the product
- • Development team - builds the product

5.1.2.1 Product Owner

The Product Owner is the member of the Scrum team charged with maximizing the value of the team's work. The product owner holds the product vision and works closely with stakeholders, such as end users, customers, and the business to cultivate and nurture a community around the product. They facilitate communication between the team and the stakeholders and ensure the team is building the right product. They describe what should be built and why, but not how.

To fulfil the role, the product owner:

- • Decides what goes into the product backlog and, equally important, what does not
- • Maintains the product backlog and orders the items in the backlog to deliver the highest value
- • Works with the team and the stakeholders to continuously improve the quality of the product backlog and everyone's understanding of the items it contains
- • Decides which product backlog items to ask the team to deliver in the current sprint
- • Decides when to ship the product, with a preference toward more frequent delivery.

The product owner may be supported by other individuals but must be a single person to maintain clarity of the vision and priorities.

5.1.2.2 ScrumMaster

The ScrumMaster is a servant leader, helping the rest of the Scrum team progress. He/she keeps the Scrum team productive and learning. He/she must have a good understanding of the Scrum framework and the ability to train others to use it. The ScrumMaster has three core responsibilities:

Coach the team

The ScrumMaster helps the entire team perform better. They help the product owner understand how to create and maintain the product backlog so the project is well defined and work flows smoothly to the team. He/she also works with the whole Scrum team to determine the definition of done. They coach the team on

how to execute the Scrum process, helping them learn and use the framework and find and implement technical practices so they can reach done at the end of each sprint.

Keep the team moving forward

As a servant leader, the ScrumMaster fosters the team's self-organization and then sees that distractions and impediments, or roadblocks, to the team's progress are removed. Impediments may be external to the team, like lack of support from another team, or they could be internal, like the product owner not knowing how to prepare a proper product backlog. They may also facilitate regular team meetings to ensure that the team progresses on its path to done.

Help everyone understand Scrum

The ScrumMaster ensures that Scrum is understood and in place, both inside and outside the team. He/she helps people outside the team understand the process, as well as which interactions with the team are helpful and which are not. The ScrumMaster helps everyone improve to make the Scrum team more productive and valuable.

5.1.2.3 Development team member

The development team does the actual work of delivering the product increment. The team is a cross-functional group of professionals who, among them, have all the necessary skills to deliver each increment of the product.

The product owner makes an ordered list of what needs to be done. The development team members then forecast how much they can do in one sprint and self-organize to get the work done, deciding among themselves who does what to produce the new product increment.

Sprints

A sprint (or iteration) is the basic unit of development in Scrum. The sprint is a time boxed effort; that is, it is restricted to a specific duration. The duration is fixed in advance for each sprint and is normally between one week and one month, although three weeks is typical.

Each sprint is started by a planning meeting, where the tasks for the sprint are identified and an estimated commitment for the sprint goal is made, and ended by a sprint review-and-retrospective meeting, where the progress is reviewed and lessons for the next sprint are identified.

Scrum emphasizes working product at the end of the Sprint that is really "done"; in the case of software, this means a system that is integrated, fully tested, end-user documented, and potentially shippable.

5.2 DevOps

The actual technical process behind implementing Dev Ops for each of the solution components is discussed in detail in the previous section (section 4). This section aims to address the high-level approach to DevOps, what the aspirational process is and where additional work is required.

DevOps itself, has always been about the fundamental approach and process, rather than the tools and technologies that underpin it.

5.2.1 DevOps Vision

The project delivery approach and underpinning tools will enable us to:

- Maximise Release Confidence
- Minimise Regression Overhead
- Reduce Development Overheads

All DevOps processes and pipeline building will have the above tenants in mind – if they are not contributing directly to one of the above visions, they are not delivering true benefit and are not required.

DevOps traditional covers the following areas, the approach of which is detailed below.

5.2.2 Infrastructure Automation

Due to the data-heavy nature of the Analytics Platforms, it is rare that production infrastructure needs to be fully torn down and replaced, however it is good practice to maintain the scripts to do so regularly to protect against disaster.

The initial state of deployed components will be recorded in ARM templates for ease of redeployment and creating initial environments, the benefit of maintaining these templates decreases as the project lifecycle continues and components are updated and evolve.

Depending on the level of test automation agreed upon, infrastructure automation can provide clean test environments to isolate individual test runs.

5.2.3 Code Automation

The vast majority of data pipelines are common, repeatable tasks and can therefore be automated, along with the surrounding orchestration frameworks. This has been demonstrated in the proof of concept and should be fully implemented throughout each of the NGBI projects.

The standard approach is to adopt a templating process, whereby the first dataset is coded manually from start to finish. The code is then made into a generic template and simple scripts used to replicate this template for all other entities.

The underpinning metadata management for these processes is central to the core application.

5.2.4 Test Automation

In general, test automation is a very good idea, but it is much harder to achieve in large-scale data processing architectures then in traditional web development applications. Ideas such as test-coverage and automatic testing-on-checkin are alien concepts for many BI developers. The cost of implementing full test coverage for a system is prohibitively expensive, it is therefore more a case of "Enough" test automation to increase development velocity and provide additional release confidence.

To summarise the approach to automated testing, it falls into three categories that have increasing levels of complexity & cost to achieve. What proportion of the whole solution should be covered by test automation, should be agreed as part of the various project implementations.

These tests automatically run each time the solution's primary codeline is changed – essentially ensuring that a given development action (addition of new datasets, changes of logic etc) have not impacted the existing system.

1. Unit Testing – *All frameworks, libraries and extensions are wrapped with a unit test. This gives us confidence that our individual functions are working as expected and protects against changes to dependencies/functionality from breaking existing code. These unit tests are lightweight, very predictable and easy to write and should be the absolute minimum of implemented test automation.*

2. Integration Testing – *This would normally be around our orchestration tools, ensuring that our various component parts execute correctly, any dependency logic is followed. Essentially ensuring that, given a certain trigger, logs are produced in the expected order. We would also cover a range of unhappy path scenarios here, ensuring our error handling logic is robust and not broken by code change. This is more time consulting to implement but has definite benefits to the delivery pipeline.*

3. Regression Data Testing – *The hardest to achieve element of automated testing within analytics platforms is to perform calculation checks after transformation & business logic have been applied. This approach would involve using the code generation metadata repository to generate test data sets covering a range of test cases. The solution is then deployed to a clean infrastructure and the tests are run and results validated. The infrastructure can then be destroyed. This gives full confidence that any changes to the solution, integrations, business logic and underpinning metadata repository have not impacted the accuracy of the system. This has an initial up-front cost to develop the testing framework and then an ongoing development overhead to maintain tests for all new datasets.*

5.2.5 Atomic Design

One of the major stumbling blocks when attempting to achieve a fast, panic-free deployment is how much each incremental deployment effects existing code.

Data orchestration pipelines have traditionally contained vast, sprawling schedules which touch all data processes end-to-end. This means that any release, even of entirely separate datasets, still include some level of risk to the existing system.

The Data Ingestion pipeline deliberately treats all data processing tasks as loosely coupled, atomic tasks. The addition of new datasets has no impact on existing code and therefore carries very little risk when deploying. This vastly reduces the risk of production releases and directly contributes to the confidence of that release and the testing overhead required.

This choice of solution design allows us to spend less effort on full test automation coverage, thereby reducing the overall development overheads.

5.2.6 Continuous Integration/Deployment/Delivery Pipelines

The other side of the common DevOps process, is the automated building and release of code, directly to the end user. The three are defined as follows:

- Continuous Integration – As soon as code is committed to the main development repository, it is deployed and automatically tested in a specific environment
- Continuous Delivery – This involves the automation of code promotion, allowing you to quickly perform code releases from test environments into UAT, Production and so on
- Continuous Deployment – This takes the above processes one step further and automatically tests and promotes code to live upon check-in.

Given the nature of analytics and the expense of full test automation coverage, a normal BI solution would never implement Continuous Deployment, except for specific, isolated elements such as orchestration frameworks.

We would endeavour to implement Continuous Integration – as discussed above we would not expect full code coverage but certainly a section of the codebase would pass through an automated test suite to provide base confidence in the quality of the release.

Continuous Deployment is supported across a sub-section of the overall solution - the code generation and deployment tools used for quickly building solutions can also be tied into a deployment pipeline. There is work and overhead in configuring this to support a multiple-environment development process and so the creation of full-coverage deployment pipelines would have to be evaluated against the overall programme budget.

5.3 High Availability / Disaster Recovery

5.3.1 Hardware Failure

Automatic recovery from hardware failure is built in to the Azure Platform will all serivces/data, hosted across multiple disks/servers. Resulting in no loss of service or interuption for end users.

5.3.2 Region Wide Failure

The process of what needs to be done in the case of a region wide failure is dependant on the technology that has incurred the disaster. For the purposes of this document technologies with a similar process have been grouped together, and then described below.

Group	Technology
1	Azure Analysis Services
2	Azure SQL DB
3	Azure Data Lake Analytics, Azure Data Factory, Azure SQL Data Warehouse, Data Science Virtual Machine, Logic Apps
4	Azure Data Lake Store
5	Data Catalog

5.3.2.1 Group 1

- Change Parameters in ARM templates to include the region for the new resource
- Deploy resource in a new Region via the ARM template.
- Restore latest backup for AAS Models

5.3.2.2 Group 2

- Geo-Restore features of SQL DB will automatically restore the latest backup in a new region.

5.3.2.3 Group 3

- Change Parameters in ARM templates to include the region for the new resource
- Deploy resources in a new Region via the ARM template.
- Deploy latest source code from GIT to the new resource
- NOTE: For Azure Data Factory, and Azure SQL Data Warehouse, connection strings to the other resources will need to change as the name of the resource will changed to reflect the new region.

5.3.2.4 Group 4

- Dependant on options described in section 4.1

5.3.2.5 Group 5

- Data Catalog has built in Geo-Redundancy and automatic failover, at no additional cost.

5.4 Data Governance

5.4.1 Lineage

The Load ID is used as primary mechanism for ELT lineage through the data lake. The Load ID and the Load timestamp are stamped on all data sources as they are moved from RAW to CURATED, providing a useful history of each data source that has been loaded in the lake.

5.4.2 Audit

Azure Data Factory is used to log audit events in a SQL DB named BI System while moving the data in the lake. The events are logged to a table that captures the following details:

Column Name	Description
Load ID	The primary key and overall global identifier of each run within the system
Execution Run ID	The data factory execution run ID
Entity Name	The name of the entity that was executed. It identifies the data source that was moved and the target within the data lake (e.g. BASE_AW_VEHICLEPOSITIONS)
Pipeline Name	The name of the pipeline that was executed
Start Time	The execution start time
End Time	The execution end time
Duration	The duration of the execution in seconds
Status	The status of the execution (e.g. Succeded)

5.4.3 General Data Protection Regulation

The General Data Protection Regulation, or GDPR, is a European privacy law superseding the EU Directive 95/46/EC, also commonly referred as the Data Protection Directive. The GDPR, due to take effect on the 25th May 2018, defines a new set of rules for privacy rights, security and compliance. It imposes new rules on organizations of all sizes and industries that offer goods and services to people in the European Union (EU), or that collect and analyse data tied to EU residents (full version of the GDPR law here).

The GDPR is structured around six principles:

- Requiring transparency on the handling and use of personal data
- Limiting personal data processing to specified, legitimate purposes
- Limiting personal data collection and storage to intended purposes
- Enabling individuals to correct or request deletion of their personal data
- Limiting the storage of personally identifiable data for only as long as necessary for its intended purpose
- Ensuring personal data is protected using appropriate security practices

Personal data means any information relating to an identified or identifiable natural person

To help the organizations to stay complacent with the GDPR, Microsoft has developed a four-step process (whitepapers can be found here and here):

- **Discover** – Identify which and were the personal data resides
- **Manage** – Govern how personal data is used and accessed
- **Protect** – Establish security controls to prevent, detect, and respond to vulnerabilities and data breaches
- **Report** – Keep mandatory documentation, manage data requests and provide breach notifications

The following sections details each one of the above steps and demonstrates how to comply with GDPR using the Azure services.

5.4.4 Discover

To understand whether the GDPR should be applied, it is necessary to identify the systems that provide and store the data, how it is collected and processed, and, for how long it is retained.

Azure Data Catalog provides the organization with a strategic platform to become and stay complacent by building an inventory of the data sources and by categorising, tagging and defining ownership on the data sources.

Azure Data Factory orchestrates and automates the movement and transformation of data. It has the capability to update the logging tables with details that can help identify the origin and destination of the personal data and when it was moved.

Azure SQL DB and Azure SQL DW have searching capabilities that are fully supported through queries.

Azure Data Lake Store can define retention policies by setting expiration dates for each data source.

5.4.5 Manage

Once the discovering phase is completed, a data governance plan should be implemented to help define policies, roles and responsibilities to access, manage and use personal data.

When using the Microsoft cloud services, the organizations are partially complying with GDPR since the cloud services are developed using the Microsoft Privacy-by-Design and Privacy-by-Default methodology. When storing the data in Azure, the organizations become the sole owner by retaining the rights, title, and interest in the data (further details on data protection here).

Azure Active Directory ensures the organization have control on who has access to the services and how the data is managed while Azure Data Catalog helps the organization to comply with the data classification guidelines by setting classification sensitivity levels on all enterprise data sources (further details here).

5.4.6 Protect

The GDPR requires that organizations take appropriate technical and organizational measures to protect personal data from loss, unauthorized access or disclosure. When hosting the data in the Microsoft cloud, the organizations are partially complying with the data security guidelines, since the Microsoft has in place security standards safer than most of the on-premises computing environments (e.g. datacenters are protected by 24-hour physical surveillance and have strict access controls).

Azure Storage Service Encryption (SSE) for Data at Rest helps the organization to stay compliant by providing a comprehensive set of security capabilities that protect and safeguard the stored data. With this feature, Azure Storage automatically encrypts the data prior to persisting it to storage and decrypts it prior to arrival.

Azure Analysis Services uses Azure Blob storage to persist storage and metadata for Analysis Services databases. It also blocks all client connections, other than the ones specified in the rules, by setting firewall rules.

Azure SQL DB and Azure SQL DW are encrypted at rest with transparent data encryption (TDE), which performs real-time I/O encryption and decryption of the data and log files. The services also limit the access to individual databases by restricting access to only authorized users through the definition of firewall and authentication rules and the management of role and object-level memberships.

Power BI restricts access to personal data by implementing row-level security and defining filters within roles.

The ngBI platform will provide the required information to the corporate AW SIEM product. This will be further defined in the Detailed Design Phase, as AW are currently transitioning to a new SIEM service provider.

Users of the new ngBI platform will use Microsoft Multi-Factor Authentication. This will be re-iterated in the detailed design phase for the Active Directory solution.

5.4.7 Report

GDPR defines a new set of standards to ensure organizations are transparent about how they handle and use personal and how they maintain documentation. The organizations processing personal data will have to maintain documentation about:

- The purpose of processing
- The processed personal data
- All parties with whom the data is shared and the legal basis for such transfers
- Organizational and technical security measures
- Data retention times

GDPR also requires the organizations to notify either the regulators or the affected data subjects when a data breach is detected. Microsoft has set a detailed Security Incident Response Management (detailed here) for the situations where they are hold some or all of the responsibility. In addition, they also created a Shared Responsibility Model that outline how they work collaboratively with the customers (details here).

Microsoft has a set of auditing tools that can help organizations to stay GDPR compliant by ensuring any processing of data is tracked and recorded (more details here).

6 Appendix

6.1 Document References

Below is are the accompanying documents to this that have either been referenced or used in the creation of this High-Level Design.

Document Name	Author	Note
NGBI Requirements Traceability Matrix	AWS / Adatis	The Traceability Matrix, created by AWS, has been modified to accommodate notes on how individual low-level requirements are supported by the architectures described in this document.
Foundations NGBI PoC Technology Deliverables	Adatis	This document contains details on all the technology deliverables from the PoC and how each element has been built and operates.
NGBI PoC Costing Calculator	Adatis	This is an Excel based costings calculator that was used to estimate the Solution Running Costs found in this document.

6.2 PoC Deliverables

Below is a list of all the deliverables from an Adatis side for the NGBI Foundations PoC, along with a note in where they can be found or how they have been delivered.

Type	Item	Note
Deliverables	High Level Design	Delivered
Deliverables	Input into Project Plan	Slide packs produced and proposed during meetings
Deliverables	Input into Test Strategy	Produced by Alan Downie.
Deliverables	Weekly Status Reports	Provided throughout project.
Deliverables	Telematics PoC	Built and demoed each sprint, deployed to AWS Azure, Source code in AWS GIT Repository
Deliverables	Pollutions PoC	Built and demoed each sprint, deployed to AWS Azure, Source code in AWS GIT Repository

Deliverables	Data Science PoC	Built and demoed each sprint, deployed to AWS Azure, Source code in AWS GIT Repository
Non-Functional	Data Lake Structure & Processing Approach	Part of HLD (See Section 3.1)
Non-Functional	Development Environment & Process Setup	Part of HLD (At individual technology level, throughout Section 3&4, at Solution Level see Section 5)
Non-Functional	Standards, Conventions, Operation Processes & Best-Practice guidance for: ADLS/ADLA/SQLDW/AAS/ADF/PBI/Data Catalogue	Part of HLD (At individual technology level, throughout Section 3)
Non-Functional	Monitoring & Support recommendations	Part of HLD (At individual technology level, throughout Section 4)
Non-Functional	Estimated Monthly Azure running costs	Part of HLD (See Section 6)

6.3 Design Standards

When building a complex business intelligence solution with dozens of components, it becomes imperative to adopt a set of designing standards to allow the solution to be scalable and compatible.

Properly constructed naming standards should identify a number of characteristics embedded in the name itself. For a modern data warehouse solution, we need to have in consideration a couple of points:

- Not all services accept special or upper-case characters. For that reason, the majority of our services are written in lowercase and without space between the different elements of the name.
- By identifying in the name the location where the service was created, we decrease the risk of creating services in different locations and not be compliant with data governance aspects.
- The solution must be available in different development environments. Whenever a service can be replicated, the environment name should be considered.

The table below details the naming standards adopted for this solution:

TYPE OF SERVICE	ABBREVIATION	STANDARD	EXAMPLE
ResourceGroup	RG	{clientname}{projectname}{abbreviation}{location}{environment}	awsngbipocrgneudev

AzureSQLServer	SQLSERVER	{clientname}{projectname}{abbreviation}{location}{environment}	awsngbipocsqlserverneudev
AzureSQLDataBase	SQLDB	{DBname}	BISystem
AzureSQLDataWarehouse	ADW	{clientname}{projectname}{abbreviation}{location}	awsngbipocadwneu
AzureDataFactory	ADF	{clientname}{projectname}{abbreviation}{location}{environment}	awsngbipocadfneudev
AzureDataLakeStore	ADLS	{clientname}{projectname}{abbreviation}{location}{environment}	awsngbipocadlsneudev
AzureDataLakeAnalytics	ADLA	{clientname}{projectname}{abbreviation}{location}{environment}	awsngbipocadlaneudev
IoTHub	IOT	{clientname}{projectname}{abbreviation}{location}{name}	awsngbipociotneuname
EventHubs	EH	{clientname}{projectname}{abbreviation}{location}{name}	awsngbipocehneuname
AzureStreamAnalytics	ASA	{clientname}{projectname}{abbreviation}{location}{name}	awsngbipocasaneuname
StorageAccount	SA	{clientname}{projectname}{abbreviation}{location}	awsngbipocsaneu
VirtualMachine	VM	{clientname}{projectname}{abbreviation}{location}{name}	awsngbipocvmneuname
AzureDNS	DNS	{clientname}{projectname}{abbreviation}{location}	awsngbipocdnsneu
VirtualNetwork	VNET	{ClientName}{ProjectName}{Abbreviation}{Location}	awsngbipocvnetsneu
BlobStorage	BLB	{blobname}	poc
AzureFunctions	AF	{clientname}{projectname}{abbreviation}{location}{name}	awsngbipocafneuname
AutomationAccount	AA	{clientname}{projectname}{abbreviation}{location}	awsngbipocaaneu
LogicApps	LA	{clientname}{projectname}{abbreviation}{location}{name}	awsngbipoclaneuname
AnalysisServices	AAS	{clientname}{projectname}{abbreviation}{location}	awsngbipocaasneu
DataCatalog	CATALOG	{CatalogName}	Catalog
App Service Plan	ASP	{clientname}{projectname}{abbreviation}{location}	awsngbipocaspneu
Public IP Address	IP	{clientname}{projectname}{abbreviation}{location}	awsngbipocipneu
Network Security Group	NSG	{clientname}{projectname}{abbreviation}{location}	awsngbipocnsgneu