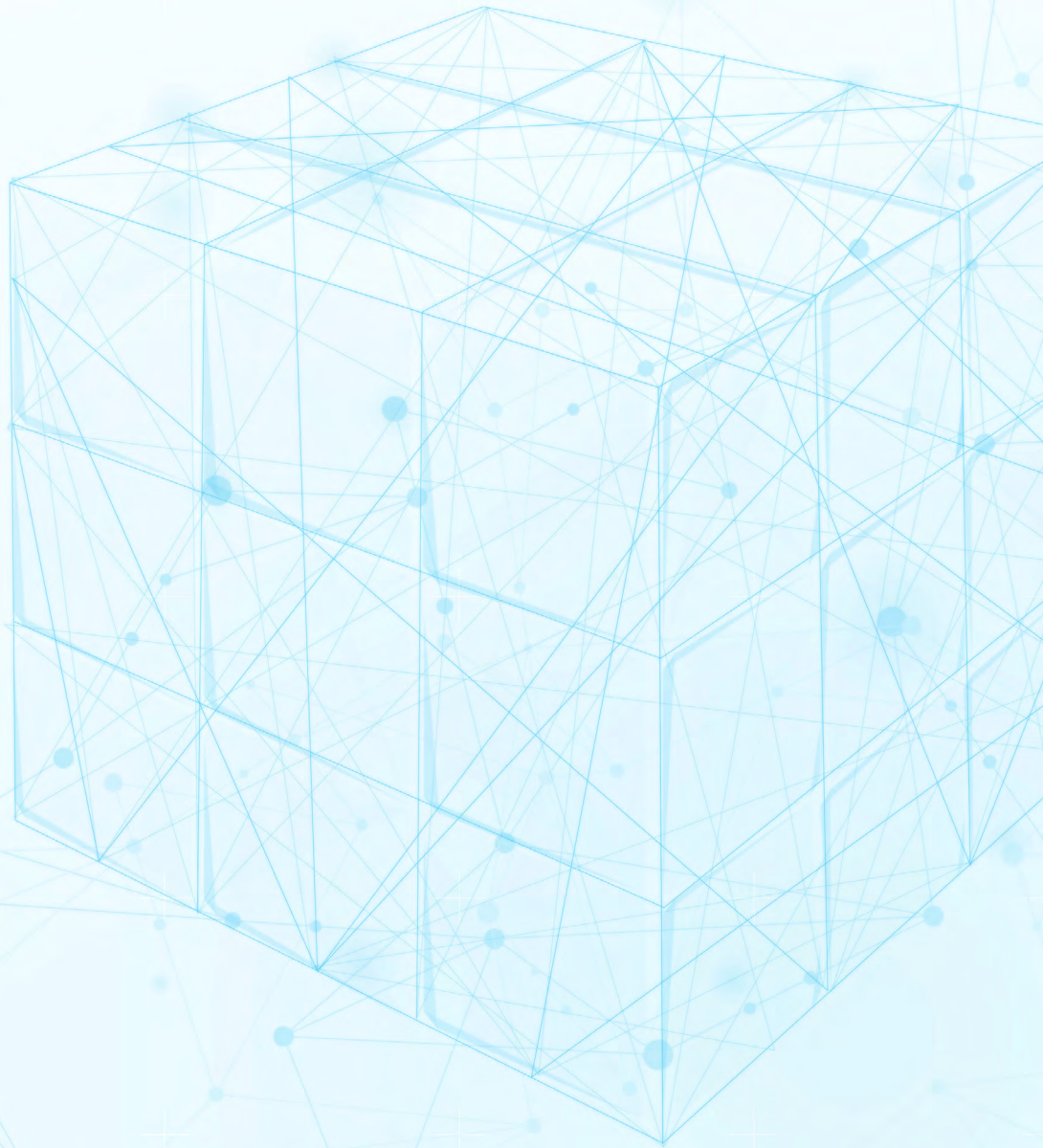
Offshore Energy Digital Architecture

NZTTP Programme

Pilot Architecture and Ontology

OEDA



Net Zero Technology Transition Programme

Net Zero Technology Centre

Technology Driving Transition

> It's proposed for the pilot that three potential interfaces to the catalogue are provided to demonstrate Foundry's suitability for OEDA, using object viewer, a carbon application and an externally hosted web application to satisfy the requirement to provide access without authentication.

Contents

1.0 Summary 4

2.0 Offshore Energy Digital Architecture (OEDA) 5

3.0 Scope 6

4.0 Background 7

5.0 Data Catalogue and Data Sharing Fabric 8

6.0 Foundry Introduction 9

7.0 Data Provider Integration 10

- 7.1 Metadata Standards 10
- 7.2 Processes 11
- 7.3 Technical Limitations 12
- 7.4 Pilot Architecture 13
- 7.5 External Catalogue Access 14
- 7.6 Internal Catalogue Access 14
- 7.7 Data Integration 15
- 7.8 Data Preparation 16
- 7.9 Data Governance 17
- 7.10 Data Pipelines and Data Lineage 18

8.0 Foundry Ontology 19

- 8.1 Object Types 20
- 8.2 Ontology Design 22
- 8.3 Object View 25

9.0 Embedding Enterprise Workflows 29

10.0 Conclusion 31

11.0 Appendix A: OEDA Requirements 32

1.0

Summary

The Offshore Energy Digital Architecture (OEDA) project is fundamentally a data sharing platform that enables awareness and access to relevant datasets, as well as the shared analytics, increased use of data across the sector to support decision making, increased use of automation, remote control technologies, and improved operational efficiency.

The Net Zero Technology Centre has partnered with InDHu, a start-up that has the principal members responsible for driving the digital transformation at Airbus, to provide a literature review and configure Foundry for the pilot in phase 2 of the OEDA project.

OEDA Report 1 - Data Sharing Landscape captured the output of an extensive literature review that defined the OEDA Requirements from a consolidated set of recommendations, best practices and lessons learned from existing implementations across a chain of eight reports in the wider energy sector, both onshore and offshore, from June 2019 to June 2022.

OEDA Report 2 – Technical Feasibility demonstrated that the OEDA requirements can be met with a high confidence similar to the UK Government definition of Technology Readiness Level 6 (TRL6) to deliver on the OEDS Data Catalogue and Data Fabric recommendations.

In OEDA Report 3 – Pilot Architecture and Ontology - an architecture for the pilot was proposed using Foundry native features, a customised experience based on the rapid application building capability utilising the Ontology using Carbon and an externally hosted web application that utilises the Platform's API. It was demonstrated that with the exception of a single OEDA Requirement that stipulated that OEDA should be based on open-source software, all other requirements, including that of the data practitioners, were met.

The intent of the pilot was also to assist industry stakeholders and consortium members in identifying what they need from a data sharing platform based on experiences of using one. This body of work has demonstrated several agile approaches to structure the catalogue using object types and therefore downstream applications that support different workflows. This can promote active discussion in this area by a demonstration of principles rather than hypothetical presentations.

The importance of Report 3 – Pilot Architecture and Ontology – is to demonstrate to industry how using an existing architecture the requirements set out in Report 2 – Technical Feasibility can be achieved and showcased.

2.0

Offshore Energy Digital Architecture (OEDA)

There are five reports in establishing a sector wide OEDA:

1

OEDA
Data Sharing Landscape

2

OEDA
Technical Feasibility

3

OEDA
Pilot Architecture and Ontology Design

4

OEDA
Potential Business & Cost Model based on Pilot

5

OEDA
Review

The first report defined and derived technical requirements for a OEDA Data Sharing Platform by evaluating existing implementations, best practices and recommendations from the wider energy sector and translated them into terms understood within the data industry. The second report demonstrated that an OEDA Data Sharing Platform is technically feasible using an example open source-based architecture to perform the evaluation. This is the third report and is based on the pilot to determine to what extent the Data Sharing Platform requirements could be met and by using it what features are likely to be required from OEDA. The fourth report examines a potential business and cost model for OEDA. The final report documents the OEDA project and provides recommendations to establish next steps.

To help determine requirements for a sector wide data sharing capability, the OEDA project will use Palantir Technologies' Foundry[1] platform along with InDHu[2] as partners for a pilot. This was primarily due the success of Foundry in the aviation sector with the implementation in Skywise[3]. Airbus was able to create an ecosystem aimed at accelerating and expanding the exploitation of aviation data across multiple parties from customers, suppliers and even competitors in the field of aircraft maintenance.

The foundation for their digital platform was Foundry from Palantir Technologies and many of the key personnel who supported the Airbus Digital Transformation are now part of the InDHu start-up. In the best traditions of the NZTC in trialling new technologies for the offshore energy sector, the OEDA project will evaluate Foundry as a pilot for the OEDA Data Sharing Platform with the expertise of InDHu in its deployment and configuration.

The purpose of this report series is therefore not to substantiate retrospectively the pilot selection. The scope is to gather existing implementations, recommendations and best practices from the wider energy sector into a preliminary set of requirements to evaluate the pilot and inform subsequent platform evaluations from other providers. Experience from the pilot will help determine and refine the proposed OEDA Requirements to support subsequent phases that will eventually lead to a tender for a Data Sharing Platform.

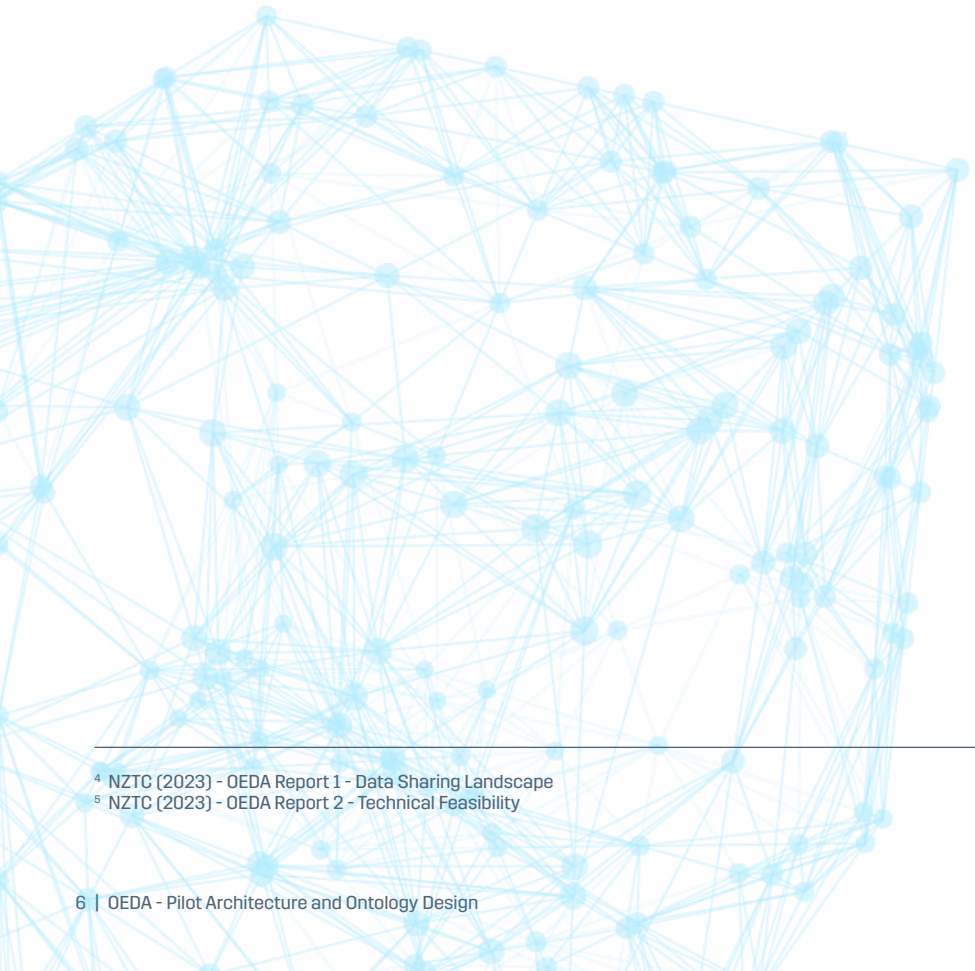
[1] Palantir Technologies (2023) - Palantir Foundry
[2] InDHu (2023) - Industrial Data Hub
[3] Airbus (2023) - Skywise | Enhance | Services

3.0

Scope

The purpose of this report is to demonstrate to what extent the pilot based on Palantir's Foundry could meet the OEDA and data practitioner requirements (included in Appendix A) established in the Data Sharing Landscape report[4] in delivering the Offshore Energy Data Catalogue (OEDC) and Data Sharing Fabric (DSF). The intent is to introduce the salient features from Foundry in the context of an industry-wide Data Sharing Platform. This will be achieved by introducing Foundry's Ontology, a key differentiator to other analytical platforms but also responsible for enabling a digital transformation with Skywise and Airbus in the aerospace industry.

To demonstrate these concepts, an example architecture and ontology design will be discussed and how it can enable the offshore industry to better explore what it needs from OEDA rather than what it currently wants. Consistent with the Phase 1 activities, notably the Technical Feasibility[5] report, a detailed design of the ecosystem based on Foundry is out of scope, instead the focus is on illustrating the key concepts and how they may support the aims of OEDA through the pilot.



[4] NZTC (2023) - OEDA Report 1 - Data Sharing Landscape
[5] NZTC (2023) - OEDA Report 2 - Technical Feasibility

4.0

Background

In 2020, the business case for OEDA (included in Appendix VII of the Net Zero Technology Transition Programme report) identified "the complexity and the scale of the challenge to integrate the data from multiple organisations, sectors, technologies, and solutions is substantial. There is a significant risk that meeting the 2045 net zero target will be impossible without investment in deploying key digital technologies in support of this target. Transformation will be excessively costly if these technologies are not deployed in a co-ordinated, collaborative way to avoid a slower more expensive transformation"[6].

OEDA is fundamentally a data sharing platform that enables awareness and access to relevant datasets, demonstrates "shared analytics platforms that are as open as possible" and promotes "increased use of data across the sector to support decision making, increased use of automation, remote control technologies and improved operational efficiency".

In August 2021, the Scottish Government awarded the Net Zero Technology Centre (NZTC) a £16.5million investment programme[7] into accelerating a range of energy transition projects to help deliver Scotland's net-zero economy. The Net Zero Technology Transition Programme was expected to enable £403billion for the economy and 21,000 jobs by 2050; it covers seven projects that have matched funding from industry:

Many of the stakeholders for OEDA include participants in the Offshore Energy Data Strategy (OEDS) Taskforce, which made two key strategic recommendations with regards to a Data Sharing Platform. OEDA is not an isolated initiative but forms part of significant movement within the wider energy sector that has produced multiple projects and at least eight related reports, both onshore and offshore, over a three-year period between June 2019 to June 2022.

Energy Hub
NZTTP Programme

Data for Net Zero
NZTTP Programme

OLTER
NZTTP Programme

Advancing Remote Operations
NZTTP Programme

Offshore Energy Digital Architecture
NZTTP Programme

[6] The Oil & Gas Technology Centre (2020) - Net Zero Technology Transition Programme - Appendix VII Offshore Energy Digital Architecture Business Case.
[7] Scottish Government. (2021) - Investing in net-zero technology - gov.scot

5.0

Data Catalogue and Data Sharing Fabric

The technical OEDA Requirements are centred on the Offshore Energy Data Strategy[8] recommendations for an Offshore Energy Data Catalogue (OEDC) and Data Sharing Fabric (DSF):

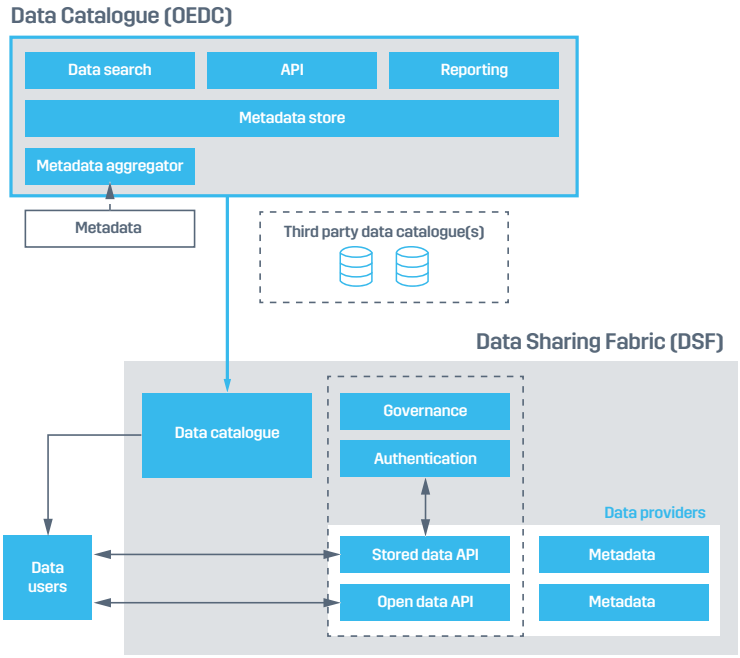


Figure 1: Offshore Energy Data Catalogue (top) and Data Sharing Fabric (bottom)

To better understand the context, an extensive literature review was conducted to derive a set of requirements based on a consolidated set of recommendations, best practices and lessons learned from existing implementations across a chain of eight reports in the energy sector, both onshore and offshore from June 2019 to June 2022. These were captured in OEDA Report 1 - Data Sharing Landscape[9] and presented in two tables.

The primary or OEDA Requirements are based on the wider energy sector (and prefixed with "E") and a second set of requirements was also proposed reflecting the expectations of data practitioners (prefixed with "D") and are both presented in Appendix A. As the offshore sector has yet to accept the proposed OEDA Requirements, both sets of requirements will be used to guide the evaluation of the pilot.

The figure above shows the general principle of how a data user accesses a resource from the data provider. The user consults the data catalogue, and if the requested resource is categorised as open, then the Data Sharing Fabric facilitates direct access to that resource at the data provider without the need to authenticate. If the resource is categorised as shared, the user needs to authenticate with the Data Sharing Fabric, which then requests Authentication and checks the Authorization before facilitating access to the resource without secondary authentication at the data provider.

The classification of the user and the request is subject to the Governance Framework, which defines the users that can connect, their level of authorization and an authentication service that delivers the required access. In practice, the Governance could be seen as a series of individual or shared Access Policies, where multiple policies may be applied to a user to provide granular control. To meet the presumed open stance, a default policy could be applied to permit permissive access to resources including (if required) no controls or restrictions at all. The "rules" that define the framework will be subject to wider industry collaboration on how to implement the principles highlighted in the OEDS Report[10].

The Fabric is intended to provide in effect Single Sign-On (SSO) to multiple data providers with one set of credentials. The workflow described is suitable where the data is directly hosted on the provider and can be downloaded or copied by the user. If the resource, however, describes an Application Programming Interface (API) to access an external data source or a Machine Learning (ML) model, the user may not have the capacity to make a local copy but will likely want to authorise a device or a compute cluster under their control, in particular to permit automated data consumption. This is not captured in the original figure, nor is this use case fully stated within the offshore sector; therefore, it has been assumed that the OEDA Data Sharing Platform needs to deliver the same intent for these types of cases.

[8] Energy Systems Catapult (2022) - Delivering a Digitalised Energy System
[9] NZTC (2023) - OEDA Report 1 - Data Sharing Landscape
[10] Energy Systems Catapult (2022) - Delivering a Digitalised Energy System

6.0

Foundry Introduction

Palantir's Foundry is an operations platform that supports the combined effort of the data, analytics and operations teams:

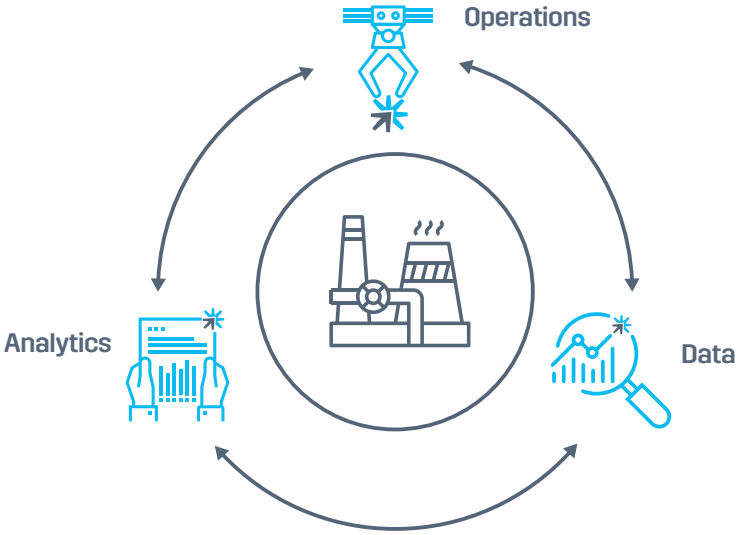


Figure 2: Foundry Overview[11]

In this context these do not have to be dedicated teams but categories - for example, the Data team could be from within the Information Technology (IT) department, or from Manufacturing, in effect anyone who has access to and manages the data of the organisation. Similarly, the Operations team represents anyone who has to use the data to make a decision affecting day-to-day operation. Such as approving a purchase request, onboarding a new employee or dealing with a customer complaint. The Analytics team could be someone analysing Seismic data or looking at employee performance and retention.

Six groups of services underpin the platform's capability:

- Data Integration
- Model Integration
- The Ontology
- Application Building
- Analytics
- Security

The Data Integration services provide a multitude of connectivity options that are scalable, integrate with Enterprise data systems and support both batch and streaming pipelines with built-in health checks. The Model Integration services integrate simulations, forecasting, predictive and Machine Learning models. The Ontology is the operational layer of the organisation as it maps the datasets and models to their real-life counterparts, in effect creating their Digital Twin. This is the fundamental component that allows applications to be constructed quickly to enable operational workflows and powers analytics without users having to worry about the Single Source of Truth or whether comparisons and calculations performed are meaningful.

Application Building powers a diverse range of users with custom and easy to use interfaces that utilises the Ontology. The Analytics services enable all users to understand the underlying data through the Ontology from no-code and low-code solutions to toolsets that empower data scientists. Finally, the Security layer enables Foundry to handle financial data, Personally Identifiable Information (PII), Protected Health Information (PHI) and a range of Government sensitive data. There is extensive documentation that demonstrates the platform supports the highest security standards in meeting OEDA Requirement E10[12].

Foundry enables an ecosystem of end-to-end Enterprise ready and robust applications to be developed within a single platform using a number of composable elements. This can therefore make it difficult to determine what tool or service is most appropriate for a particular use case. The subsequent sections define an example architecture that utilises some of the features to meet the OEDA Requirements.

[11] Palantir Technologies (2023) - Platform Overview
[12] Palantir Technologies (2023) - Security

7.0

Data Provider Integration

The Offshore Energy Data Catalogue (OEDC) and Data Sharing Fabric (DSF) are part of the same ecosystem with data providers and therefore will be influenced by their technical limitations but also metadata requirements, standards and processes which are currently not agreed across the offshore industry.

7.1 Metadata Standards

The wider energy sector has a number of overlapping recommendations in terms of the required metadata but in the absence of a consensus, it is recognised that the minimum requirements cited[13] are the Dublin Core[14] standards. This is supported by a recent decision in the onshore energy sector where Ofgem's Data Best Practice[15] reinforces the need to support the Dublin Core metadata standard. The most basic set was incorporated by the Internet Engineering Task Force (IETF) Request for Comments (RFC) 2413[16] standard as illustrated below:

It is proposed for the evaluation of the pilot to use these 15 elements, which have been broadly grouped within the RFC as relating to the Content, Intellectual Property and Instantiation (in effect when materialised at a given time and place). There is no technical limitation within Foundry on the number of metadata elements or the complexity in their hierarchical structures, therefore the RFC is a good basis for the pilot evaluation.

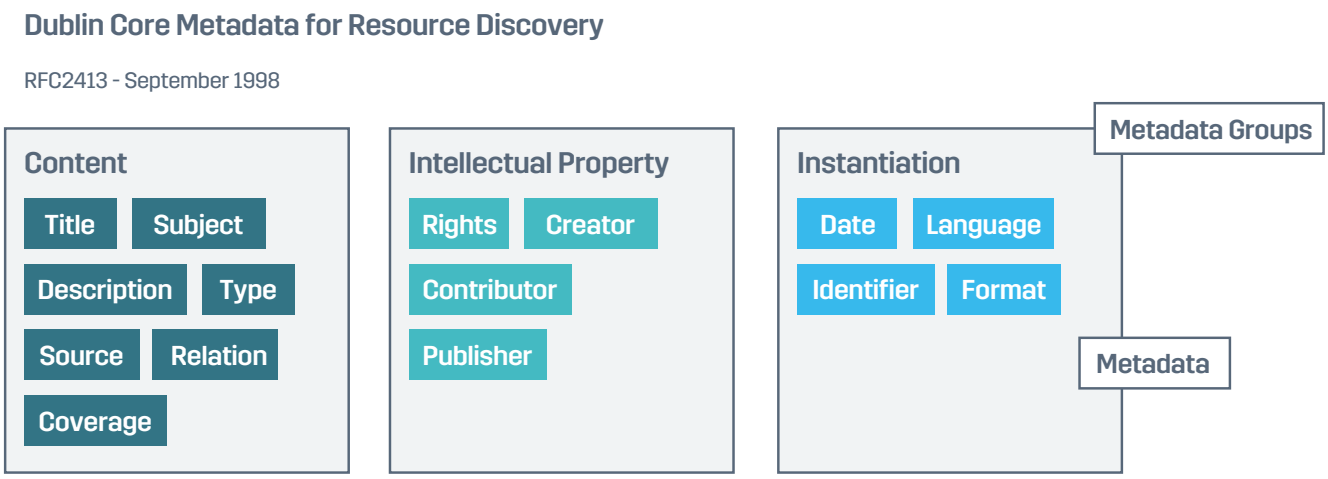


Figure 3: Dublin Core Metadata in Metadata Groups

[13] NZTC (2023) - OEDA Report 1 - Data Sharing Landscape
[14] Dublin Core (2023) - Dublin Core Metadata
[15] Ofgem (2023) - Decision on updates to Data Best Practice Guidance and Digitalisation Strategy and Action Plan Guidance
[16] Internet Engineering Task Force (1998) - RFC 2413 - Dublin Core Metadata for Resource Discovery

7.2 Processes

Technical solutions tend to focus on the end state as being static, in this instance a data catalogue. However, it is important to recognise that there are multiple workflows associated with the lifecycle of a data catalogue entry. There is the process of creation, modification and removal but as these actions can be destructive, there is also a need to define and manage who can perform these actions and what if any authorisations are required. Removing a widely used data asset without notice is likely to have a significant detrimental impact and undermine the objective of greater industry collaboration. This highlights another need in how the community is notified of such changes, in particular if they have consumed the data previously or are doing so actively.

Much of the focus from the Data Sharing Landscape has been on the infrastructure elements of a catalogue and DSF, without defining how these would be used in practice. The pilot provides an opportunity to explore these concepts and in the absence of any agreed practices, the intention is to show how these workflows could be codified within Foundry.

7.3 Technical Limitations

The two key technical interface requirements are:

Automated metadata transfer

Authentication and authorization between the fabric and the data provider

The novel component in the OEDS defined data catalogue is the inclusion of a metadata aggregator. The technical burden in the integration between the two systems can have potentially two extremes; the first is that there are no changes made to the data providers. In this instance the aggregator needs to support a variety of interfaces to extract the metadata from a data provider provisioned API, to accessing a portal securely and ingesting an XML file, to web scraping using a bot and performing complex post processing akin to web crawlers used by search engines.

The other extreme (adopted by the Ice Breaker One[17] and advocated by the EDTF approach) is to put the burden on the data provider, in either constructing an API to a set standard or deploying a metadata depositor - an automated means of translating the data provider's dataset format and metadata into a format compatible with the catalogue at the data provider's technical expense.

As there is currently no agreed data provider technical interface, the proposed architecture will accommodate most use cases for the purposes of evaluating the pilot.

[17] Icebreaker One (2023)

7.4 Pilot Architecture

The following architecture demonstrates that it is likely all but one of the 10 OEDA Requirements can be met using just native Foundry features. The exception is Requirement E3[18] which requires the use of open-source software, whereas Foundry is a commercial product using proprietary integrations of open source components. To mitigate concerns regarding access to data and workflows, Palantir has taken steps to make Foundry accessible from other platforms and improve interoperability[19].

The requirement also stipulates support of the Presumed Open principles, one of which requires data that is considered open to be accessible without authentication. It is currently not possible to access any Foundry element without authenticating given its security posture. The proposed architecture uses the interoperability features to provide an alternative whilst meeting all of the other OEDA Requirements:

There are potentially two general approaches to adopting Foundry; the first is to host the entire ecosystem (like the Skywise implementation) within the platform - this not only means the users and data providers, but also their processes and workflows. Many of the OEDA and data practitioner requirements can be met with just native Foundry features. The second approach recognises that not all data providers may wish to host their data within the platform or may wish to maintain a presence outside of it. In both cases, there will be a group of users who will wish to access the data catalogue and the open data without authentication.

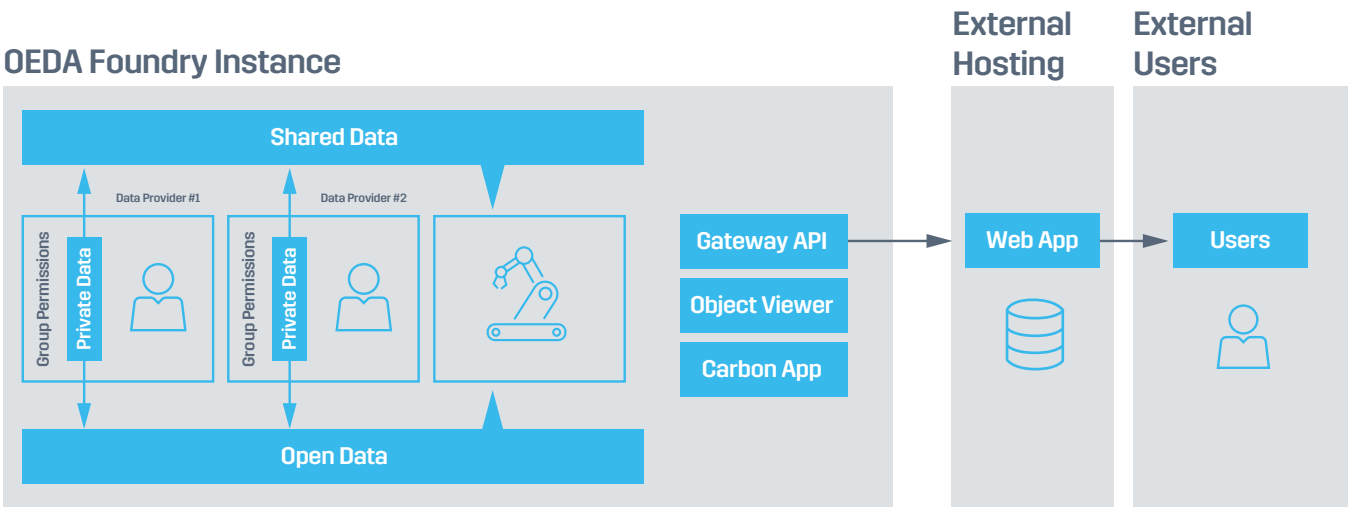


Figure 4: Potential OEDA Architecture based on Foundry

[18] OEDA shall be based on open source software and open standards. It should facilitate the Presumed Open principle.
[19] Palantir Technologies (2023) - Interoperability

7.5 External Catalogue Access

The figure above shows access to the catalogue without user authentication can be met with an intermediary external application, which authenticates with the Foundry Gateway API using OAuth2[20] (and also demonstrates that it satisfies Requirement D2[21]). The application could be web-based (as per the figure) or any other publicly accessible application such as a Microsoft Power BI dashboard similar to the approach adopted by the Office of Rail and Road in presenting Passenger Rail Performance[22]. The Foundry API provides access to all of the data on the platform through a unique Resource IDentifier (RID) as well as programmatic access to the ontology and its objects, links and actions. These terms are explained in greater detail in subsequent sections but in effect access to all the elements needed to host an external data catalogue that does not require authentication to access.

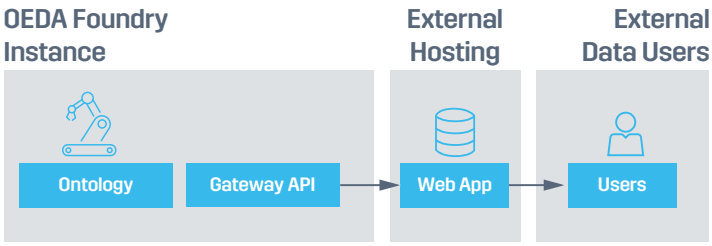


Figure 5: External User Access

7.6 External Catalogue Access

For users within the Foundry ecosystem, either in the OEDA instance or another Foundry instance, there are two proposed solutions to interface with the data catalogue. The first is the Object Viewer[23] accessed through various applications but typically Object Explorer, which is a native Foundry application that allows easy exploration of the catalogue. Whilst it is possible to manage the life cycle of each entry and associated processes with just the viewer, it requires some knowledge of the ontology and doesn't fully replicate business processes. To support users with minimal training with workflows that match the needs of organisations, Foundry encourages application building based on the ontology through its Carbon[24] service:

The key advantage of a Carbon application is that it can abstract away all of the other Foundry features and can be used to combine multiple applications built on the ontology into a single bespoke view. In practice this means most of Foundry could be hidden away from particular users providing an uncluttered and dedicated experience in accessing and / or managing the catalogue.

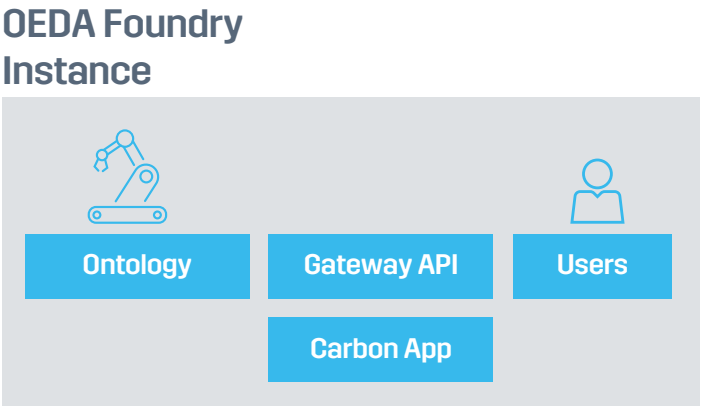
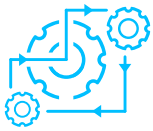


Figure 6: Internal Catalogue Access

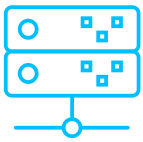
[20] Palantir Technologies (2023) - Writing OAuth2 Clients for Foundry
[21] OEDA shall support the use of long held security tokens including but not limited to client and server-side certificates - mutual Transport Layer Security (mTLS) with Hardware Security Modules (HSM) and / or rotated authentication tokens (i.e., OAuth 2.0 / OIDC).
[22] Office of Rail and Road (2023) - Passenger Rail Performance
[23] Palantir Technologies (2023) - Ontology - Object Viewer
[24] Palantir Technologies (2023) - Application Building- Carbon

7.7 Data Integration

Subsequent sections define what the ontology is, how it is used and present examples / mock-ups of the object viewer, Carbon application and the external web application. They are all predicated on the underlying data supporting the ontology, which can be integrated from data providers in broadly three different ways:



Automated data integration using agents[25] running within the provider's network or automated ingestion, pipeline creation and even ontology creation with HyperAuto[26].



Using direct connections to a data store (e.g. database, file sharing protocol or even Microsoft's Sharepoint).

Using a messaging client for streaming data[27] or REST APIs for batch[28] data.

The Data Catalogue is however intended to capture metadata about the data; there is no requirement to host the data itself, although Foundry does provide the features to both understand and utilise the data in decision making. All three methods permit the copying of data as well as copying the metadata using automated means and therefore also meet the metadata aggregator requirement of the data catalogue (E1[29]). Automated ingestion of metadata will require some co-ordination with the data providers depending on the format and type of data.

The most effective tool for data integration within Foundry is HyperAuto - a public example is the Palantir deployment as described by BP's former Chief Executive Bob Dudley in how two Palantir Forward Deployed Engineers integrated many of BP's disparate systems and provided greater visibility over just a weekend[30]. HyperAuto is designed not only to copy the data but also apply typical transformations (in effect cleaning and organising it) as well as creating an ontology automatically based on the context of the data. The second most effective method of data integration is the use of an agent, which is controlled by Foundry and provides a secure managed and highly available mechanism of data extraction.

[25] Palantir Technologies (2023) - Data Connections
[26] Palantir Technologies (2023) - Data Integration - HyperAuto: Software-Defined Data Integration
[27] Palantir Technologies (2023) - Data Integration - Streaming Sources
[28] Palantir Technologies (2023) - Data Integration - REST APIs
[29] OEDA shall support the OEDS defined Data Catalogue.
[30] Palantir Technologies (2023) - YouTube - Operational AI for Critical Institutions | Palantir CEO Alex Karp at CERAWeek (at 08:00)

7.8 Data Preparation

All three integration approaches would materialise the assets (whether it is the data or the metadata) in a private space for that organisation (i.e. data provider).

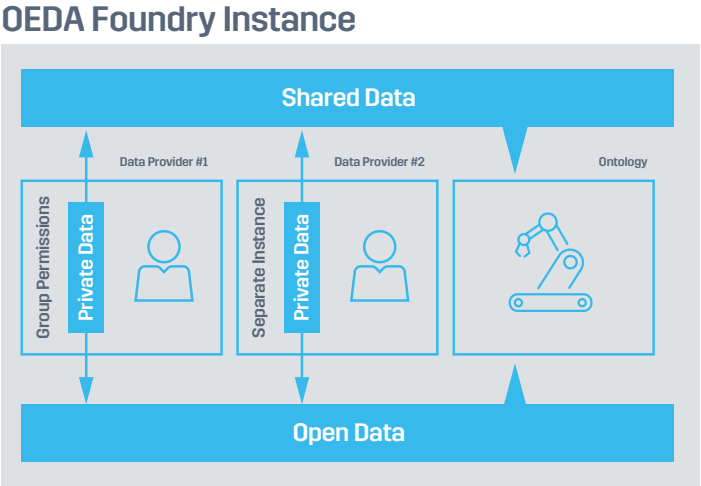


Figure 7: Data Provider Onboarding

The figure shows two data providers with a presence on the OEDA instance of Foundry, the first is an organisation onboarded directly onto the platform with a dedicated and private segment to operate in but ultimately managed by OEDA. The second represents a separate Foundry instance using its cross-organisation collaboration[31] feature.

Typically, most data providers will clean their data to remove Personally Identifiable Data (PID), commercially sensitive data or enrich the dataset to provide the proper context before transferring them on to a Data Sharing Platform. These are often manual operations that require handling potentially multiple intermediary files and ensuring the labels and locations are appropriate to prevent confusion in the future. The same intent can, however, be delivered within Foundry with fewer operations and better traceability.

Consider a scenario where an industry or a group of companies collaborate to reduce fraud involving employees or third-party contractors. An analysis conducted without any PID will make it difficult to determine which people to contact to try and understand any anomalies that have been detected. The typical approach is to split the data (with and without PID) with a look-up table that maps an individual to a non-descriptive but unique hash. The outcome of the analysis provides the hashes of interest, in which the data owners can use the lookup table to map to a real person. This has proven to be difficult to execute as it requires the handling of multiple datasets and organising permissions such that the right people have access to the right file.

Within Foundry, three native features are offered along with a built-in organisation workflow that ensures controlled access. Sensitive data can be encrypted or hashed directly on a single column or a whole dataset such that only restricted personnel can view and manipulate the data using the Cipher[32] tool. This is enforced using the approvals[33] workflow, which permits users of the data to request access but also the checkpoint[34] tool, which even if a user has prior permission must still document why they are accessing the dataset. With Cipher, the data provider can accelerate their workflows for data release by ensuring their equivalent of a data officer can approve the release of a dataset with the security controls in place. There is also a built-in Sensitive Data Scanner[35] tool to assist data providers catch unexpected sensitive data.

In the fraud detection example, the data providers can encrypt the relevant columns with Cipher and make it available in a shared space. This dataset can be combined, manipulated and analysed to produce an end result, where the data provider is the only organisation that is able to decode the sensitive data contained in the output. This, therefore, removes the need to produce and manage multiple intermediary files.

[31] Palantir Technologies (2023) - Security - Cross-Organisation Collaboration
[32] Palantir Technologies (2023) - Security - Cipher
[33] Palantir Technologies (2023) - Security - Approvals
[34] Palantir Technologies (2023) - Security - Checkpoints
[35] Palantir Technologies (2023) - Security - Sensitive Data Scanner

7.9 Data Governance

The pilot architecture shows the Landing Zone[36] of the data as a Private Space, which depending on the Governance framework from the Data Sharing Fabric would set the default access policy. Foundry provides several features to enact a Data Governance framework from basic controls such as users and groups[37], collaborative features such organizations and Namespaces[38] and advanced features such as Markings[39].

Typical permission-based control could be inherited through the Single Sign On (SSO) capability in Foundry from host organisations. These can be applied to different locations within the Foundry from where the data is stored to object types and objects in the ontology. It is proposed that each data provider is set up as an organization for the pilot to enable a dedicated Private Space to prepare their data as described in the previous section. In addition, to enable any pre-processing it is proposed that the provider belongs to their own Namespace but also two additional namespaces: open and shared. The Namespace structure permits projects to be created where ultimately the provider's data could be hosted.

These controls permit the Governance framework from the Data Sharing Fabric to be created and enforced. In maintaining a living data catalogue, there will be occasions where sensitive data may arise that requires additional controls or specific cross-collaboration controls. The features described thus far are location based, and depending on where the data is, the level of access is granted. Foundry also permits controls that can cross multiple boundaries such as Projects, Namespaces and Organizations in effect the constraints travel with the data through the Markings feature.

A potential use case could be exploring the use of a new and novel data source, which is then limited to 10 specific users from say 10 companies. Location-based controls mean that all derived data must be hosted in the same location and accessible by all 10 users. This would discourage using the data in other company confidential use cases to determine potential benefits.

Markings travel with the data, such that any derived data inherits the same constraints. So, if the derived data is stored in a company specific Namespace, the user can meet both sets of constraints. The union of permissions ensures that the organisational controls keep all non-company users out and the markings controls keeps all unauthorised users out. A combination of these features satisfies the governance and the authorisation components of the Data Sharing Fabric in Requirement E2[40].

The union of permissions ensures that the organisational controls keep all non-company users out and the markings controls keeps all unauthorised users out.

[36] Stax.io (Feb 2023) - What is a Cloud Landing Zone?
[37] Palantir Technologies (2023) - Security - Users and Groups
[38] Palantir Technologies (2023) - Security - Organizations and Namespaces
[39] Palantir Technologies (2023) - Security - Markings
[40] OEDA shall support the OEDS defined Data Sharing Fabric.

7.10 Data Pipelines and Data Lineage

At the highest level, Foundry enables multiple stakeholders or groups of users to work in a dedicated space using organisations, which could also include access through Single Sign-On (SSO). To migrate data from the private space to a location where either open or shared data resides, Foundry best practice is to use a data pipeline. These are created with either the Pipeline Builder application (no-code) or Code Repositories which is designed more for data engineers. In general, pipelines take single or multiple original datasets, transform, filter and aggregate them into an output dataset, however even if no changes are made the use of pipelines is recommended.

Pipelines are the foundation of creating automation, permit data health checks and scheduled updates; but their most significant benefit is that it permits Foundry to create a data lineage graph across organisational and usage boundaries (figure 8).

Each node in the figure represents the name of the dataset usually prefixed with the relative folder name where the lineage graph is stored; the edges (or links) typically represent a data pipeline that links the datasets. The transformations between nodes are not visible in the data lineage graph although a variety of metrics and features can be displayed with the node colour[41] using the key. If Foundry best practice is followed, then a group of transformations or a project should utilise the Repository[42] feature (or folder name as a back-up) and much like Version Control Systems (VCS) used in software development, the user can work in an agile manner but have the ability to revert changes.

By viewing the data lineage as a function of repository name, it provides valuable insights into how the data is being used. The figure illustrates that the output from the data provider is a dataset in a shared location and is then used in three subsequent repositories across two organisations. The Start-up appears to have two projects that utilise the shared data and appear to merge a company specific dataset to enrich the output. Although the access to these datasets is restricted to their respective organisations, these parameters (such as the dataset name) are visible to all users of the platform.

The data provider can quickly determine how their data is being used but it also allows collaborators to determine if they are likely making the same transformations in isolation and therefore, it may be better to do so once and then share the resultant output. There are other metrics visible such as row count, number of files, build time, frequency and duration - all of which can be used to determine the level of engagement with the data. In contrast, the North Sea Transition Authority (NSTA) receives tens of millions of API calls every year but without this level of visibility on who is using the data, how much, how often and for what purpose. The features discussed support the OEDA Requirements E6[43], E8[44] and E9[45] as well as the data practitioner requirement D1[46].

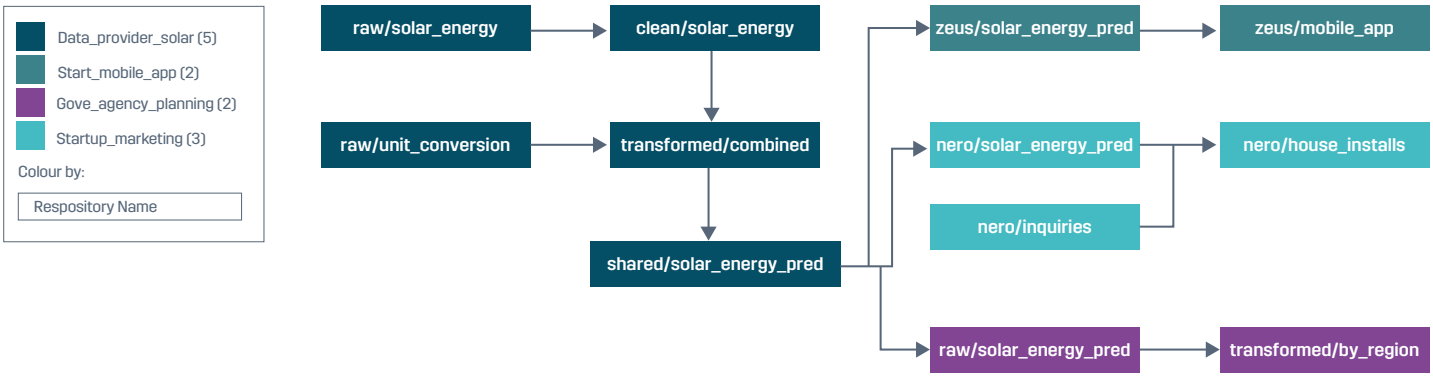


Figure 8: Data Lineage graph illustration

[41] Palantir Technologies (2023) - Data Lineage - Node Coloring
[42] Palantir Technologies (2023) - Data Integration - Branching
[43] OEDA shall support metrics regarding the data.
[44] OEDA shall support a mechanism to enable users to provide direct feedback to Data Providers.
[45] OEDA shall display lineage or provide the means to define a lineage between datasets. OEDA shall support datasets to be related using attributes.
[46] OEDA shall support the use of internal and external repositories for dataset documentation, context, data samples, API definitions and other assets.

8.0

Foundry Ontology

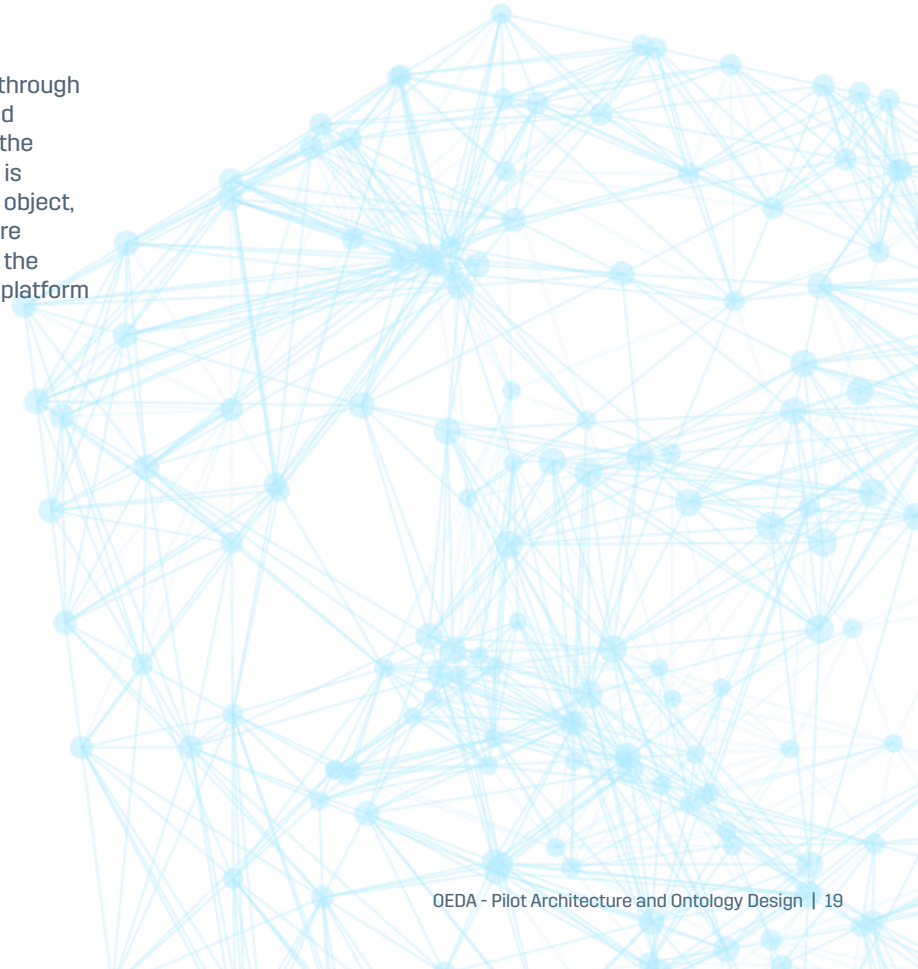
The ontology is the operational layer of the platform and is intended to allow users to create business representations of their data. It uses the concept of an object type to describe a class or type of object that maps to its real-life counterpart. For example, an object type for a car, will consist of properties that describe the number of wheels, the colour, its valuation and other attributes deemed relevant in the business context. An object is therefore a single or particular instance of an object type.

Each object type is mapped from a single or multiple datasets (referred to as the Backing Dataset), where for a tabular source, each column would map to a property of the object type and each row represents a single instance or object. Relationships between objects, for example where a car is serviced (modelled as a Garage Object Type) are referred to as object links.

Foundry encourages adding a semantic layer to the data through the use of object types to exploit the platform's automated exploration and visualisation tools. For example, if one of the properties of the car object type is a parking location and is declared as a geospatial property, then when viewing the object, a map will be automatically displayed. If two car objects are compared, instead of providing a numerical difference for the location property, it will calculate a distance because the platform is aware of the semantic nature of the property.

A fundamental difference between a dataset and its ontology representation is under operational contexts most businesses want controlled changes during the lifecycle of an object; in practice a minimum of controlling and recording the who, recording the when, the where and the how. One method to change the ownership property of a car is to manually edit the backing dataset but then it is open to errors, it doesn't follow the business process and potential for multiple users attempting to perform the same operation.

The ontology allows users to define what properties can be changed and how through actions types. These can be combined with the application building layer to create user friendly forms as part of work flows that mimic business processes. For example, on a car it is expected to update the mileage property but although unlikely and rare, it is also possible that the unique serial number (or VIN) could also require a change. The former could be enabled for most users, whereas the latter could be restricted to a subset of users or require a workflow that includes Checkpointing to ensure the correct evidence is logged prior to making the change.



8.1 Object Types

The word ontology is defined as a set of concepts in a defined area that shows their properties and the relationships between them. It is used as a way to define a Digital Twin, an abstraction that captures the salient properties and their relationships or sometimes referred to as providing the Semantic layer. There are other definitions within the sector such as from the OSDU Forum[47] and the Open Energy Platform[48].

The approach in Foundry reflects the convergence of three computing trends over the past 50 years. Since the 1970s, database administrators have attempted to construct a data model that represents how a business operates through the use of tables and their relationships between them, similar to constructing the backing dataset for an object type. The second trend also started in the 1970s is called Object Oriented Programming (OOP) in software development - it was recognised that existing data types such as integers, floats and strings did not accurately model real-life objects and their properties.

The intent of OOP was to create a programming representation of real-life objects through classes (referred to as object types in Foundry), where a single instance of a class is also referred to as an object. The third trend in response to increased cybersecurity concerns was the popularity of type safety, in effect when a variable in a program is defined as an integer but receives a float (a decimal number) the program can be made aware with an appropriate response. A prominent example is the open-source language TypeScript[49], which took one of the most popular languages in the world - JavaScript - and added support for type definitions.

The illustration shows how a programming language or spreadsheet could be used to model a business scenario. The columns show a single object representation, an example of comparing two objects, examining multiple objects and a brief comment on the semantic representation. The first row shows a number as a single object and most computing languages will permit two numbers to be compared with subtraction. In the field of data science and data analytics, there are tools that will automatically generate summary statistics and plots - referred to sometimes as Automatic Exploratory Data Analysis (EDA) or data profiling depending on context (based on recognising the object to be a number). In this example, if the object is attempting to reflect a printed ticket which has a queue position, then many of the operations performed and plots produced have no real meaning such as the sum function for a collection of queue positions.

It is important to note that these automated tools have no awareness that the number actually reflects a queue position. Traditionally, to prevent numerical operations from occurring on non-numerical entities an appropriate data type is set, typically in this scenario a string. The functions to compare strings are different to that of numbers and there are limited statistical properties or plots to produce. Although this approach mitigates accidental summation of a queue position, it still does not reflect the queue position behaviours, for example sorting the strings in alphabetical order will put the Ticket “10” before the Ticket “2” and still therefore doesn’t reflect the real-life ticket in this business context.

The third row reflects what is meant by a semantic type, in that through some customisation or the use of OOP we create the type Rank. This provides certain methods and functions that will sort the data based on a predefined map (useful when ordering is non-alphabetically e.g. a status such as Platinum, Gold and Silver) and prevents some operations (such as addition) but permits others. Although this is a closer representation of the ticket, it only has a single property - the queue position. In practice our real-life ticket may have a date and time printed, a ticket station number and when the number is called a backend system could log both the start and end time.

The final row therefore reflects an object type, which consists of the queue position number but also a range of other related properties. When comparing two objects of the ticket type, it is possible to get meaningful comparisons such as what was the difference in duration in minutes. This extends to multiple objects, where a histogram could display the duration distribution, a timeline constructed using the start and end datetime fields and a heatmap of locations to show which station had the greatest number of people. The representation also prevents inappropriate functions from being applied, for example the mean of the duration will be calculated but not the mean of the queue position.

[47] The Open Group (2023) – OSDU Forum
[48] The Open Energy Family – The Open Energy Platform
[49] Microsoft (2023) - TypeScript: JavaScript with Syntax for Types

8.1 Object Types

Object types are created using the Ontology Manager[50] and maps fields from a backing dataset, their semantic representation and the links to other object types. In traditional database design, considerable effort is expended on the data model prior to implementation due to constraints in performing in what would be considered routine operations such as renaming, adding or deleting a column. Therefore, during the lifetime of the database there is significant resistance to any schema changes.

In contrast, it is recognised within Foundry that users may not be fully accustomed to the concept of an ontology and it is difficult to determine if the choice of object types has the right set of emergent properties without having trialled the design. Foundry, therefore, supports multiple object types to be created from the same dataset and the object view for a given object type is also fully versioned controlled supporting a more agile approach to determine the most effective structure.

The following figure illustrates what these terms mean in practice:

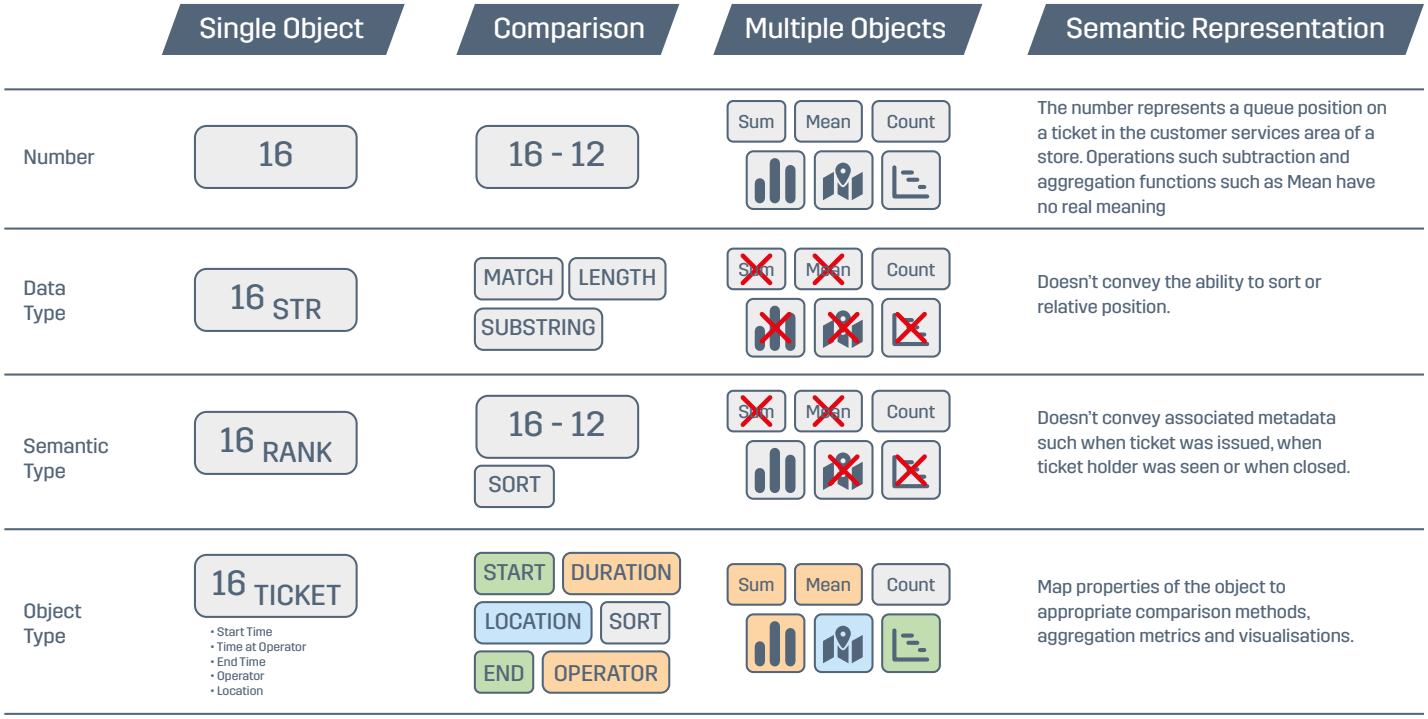


Figure 9: Evolution of Semantic Representation

[50] Palantir Technologies (2023) - Ontology - Ontology Manager

8.2 Ontology Design

There are two key challenges in designing the ontology for the OEDA Data Sharing Platform; the first is an understanding of the metadata structure applicable for the datasets expected across the entire offshore sector. The second is an understanding of the operational workflows expected by data users and data providers in maintaining the catalogue, the automated metadata aggregation and the Data Sharing Fabric. As stated previously, in the absence of a consensus on the minimum metadata, the Dublin Core set from 1998 will be used as shown below. It should be noted that the OEDA Data Sharing Landscape report identified that many of the recommendations from the wider energy sector may not reflect the types of data assets expected within the offshore industry. Similarly, the workflows discussed have focussed on the process to ultimately determine if a dataset can be released and its status in terms of open or shared and not the examples stated above.

The pilot offers the opportunity to explore and develop the thinking around these other aspects of owning and maintaining the OEDS defined data catalogue and Data Sharing Fabric. The approach to utilising the platform cannot therefore follow a typical Foundry implementation approach as the inputs have not been sufficiently defined. The intention is therefore to illustrate the capabilities of the pilot in this field and showcase a starting example with consortium members and industry stakeholders in order to seek feedback and refine the design.

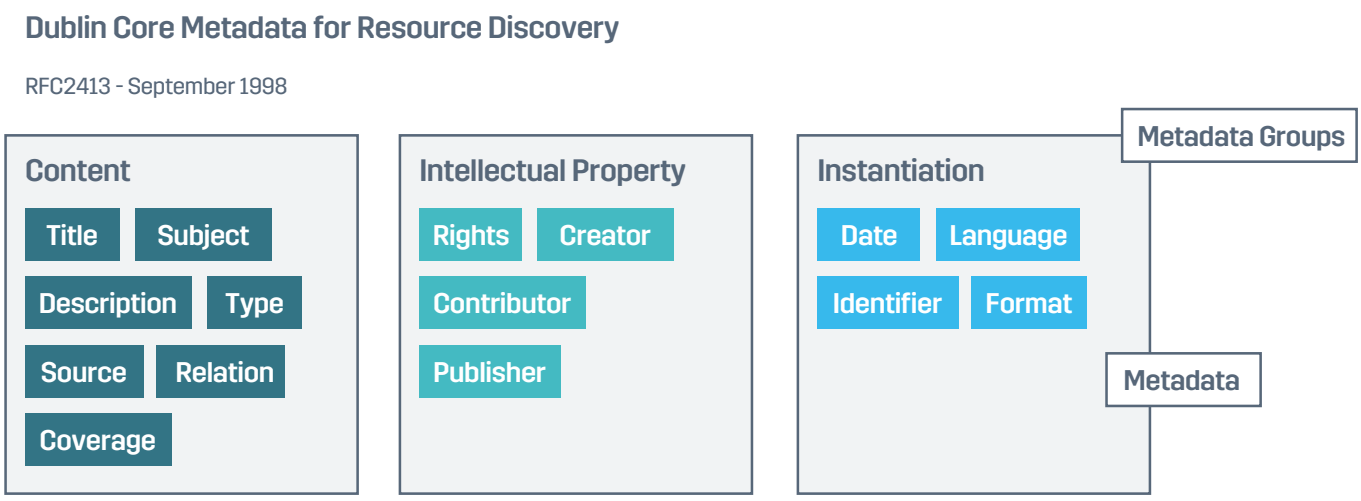


Figure 10: Dublin Core Metadata in Metadata Groups

The most basic approach is to treat the catalogue as a tabular dataset and in effect have a single object type with all the metadata elements above as properties. Within the metadata listed, it is also possible to identify other individual entities, which may be worth modelling as their own object types; one example is that of publisher or in our case the data provider:

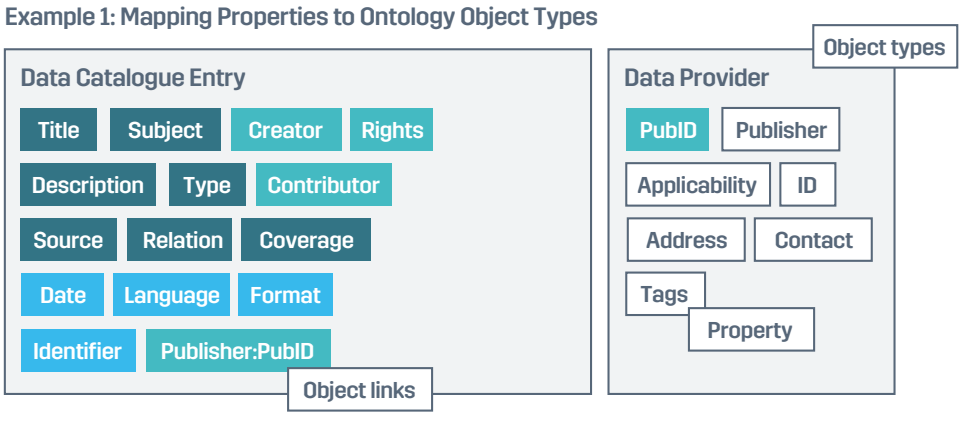


Figure 11: Data Catalogue and Data Provider Object Types

8.2 Ontology Design

The figure shows two object types, one is a catalogue entry and the second is a data provider, which are linked together using a common identity. This permits additional properties to be captured for the Publisher in such a way to support operational workflows. For example, we can potentially browse and filter a set of providers based on tags providing an alternative view into the data catalogue but also from an operational perspective maintain an independent view on contact details and associated personnel. These fields would clearly be inappropriate to host in a public data catalogue and therefore creating a separate object type is both semantically and operationally useful.

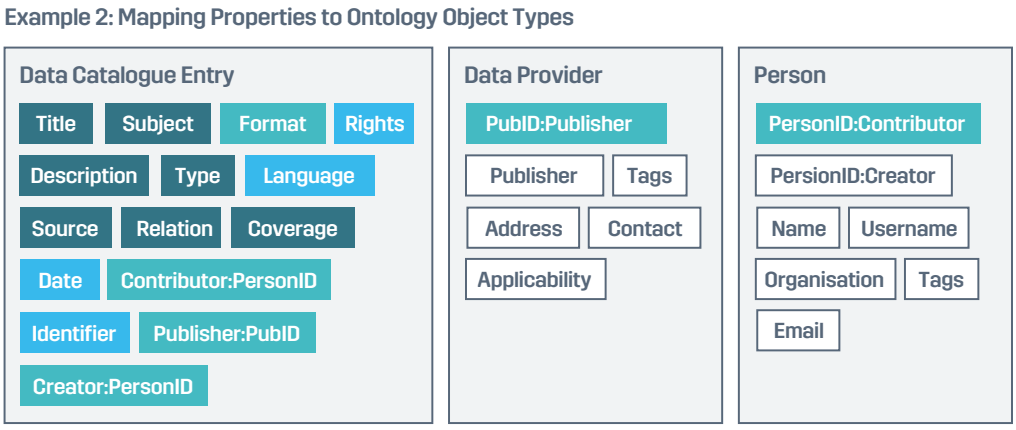


Figure 12: Extending Ontology Design with Person Data Type

The same approach can be used for the contributor and creator fields as they are in general natural persons or can be expanded to be both natural persons and organisations. The figure above shows a similar approach, and again facilitates the exploration of the catalogue through an author or researcher. It is recognised that for most instances, the creators may not be on the platform but the creation of a separate object type with a link enables the user to utilise the object links to find related datasets in this instance by the same creator or contributor.

OEDA Requirement E4[51] and related Requirement D6[52] recognise the need to have a structured approach to metadata for example using a taxonomy and using potentially a hierarchical layout. This can be implemented through the choice of backing dataset. For example, suppose the wider community agrees on a set of keywords to use, these can be imported into the platform and used to back a property in the object type definition:

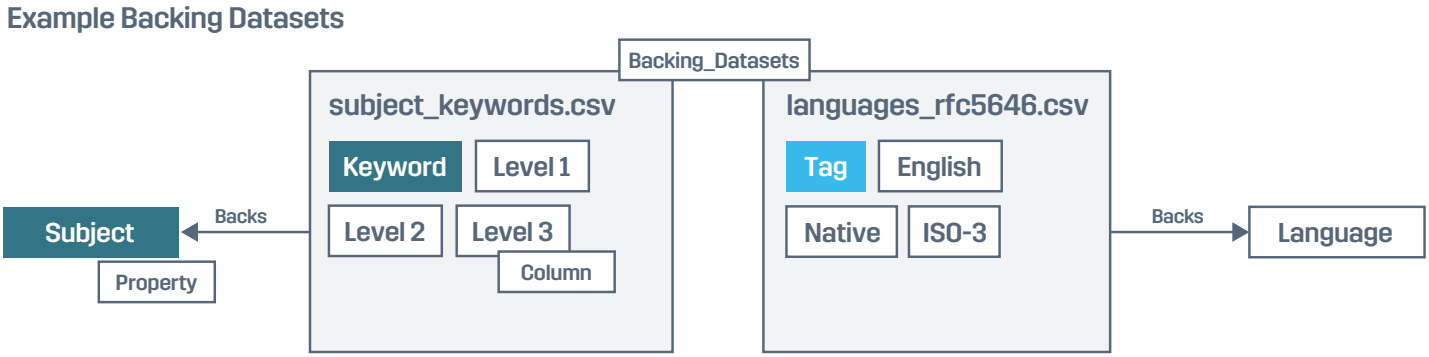


Figure 13: Effective use of Backing Dataset for a Object Type Property

[51] OEDA shall support a customisable set of attributes to act as metadata and have the means to define differing levels of priorities and controls.
[52] OEDA shall have the means to support a variety of metadata formats (beyond the current attribute-oriented needs).

8.2 Ontology Design

This feature can enforce the Taxonomy through a predefined set of keywords that can be used and control the number of tags that can be applied. For example, a workflow that triages data release requests, may have the status new, assigned, hold and release; these could be defined via a backing dataset and manifests itself to the user as a dropdown menu with four options. Coupled with the Foundry Rules[53] feature, automated alerts and notifications could be generated or workflow alerts triggered.

The second example mitigates an observation from the Energy Data Centre and other data catalogues where the language field is free-form text resulting in many but related terms such as: English, english, eng, eng/GB and en-GB. Using an imported dataset based on an international standard as the backing dataset for an object type field restricts the potential inputs to a standard set.

Once the objects are being used the Ontology Manager also contains a usage screen that provides visibility on how and where they are being used (figure 14).

In addition to the data lineage graph capability, this provides additional context on object type popularity and the applications that have been built upon them supporting Requirements E6[54] and E8[55]. Co-ordinating object type creation by duplicating existing types and modifying them allows for an agile development process to determine the most effective use of the platform to meet business needs.

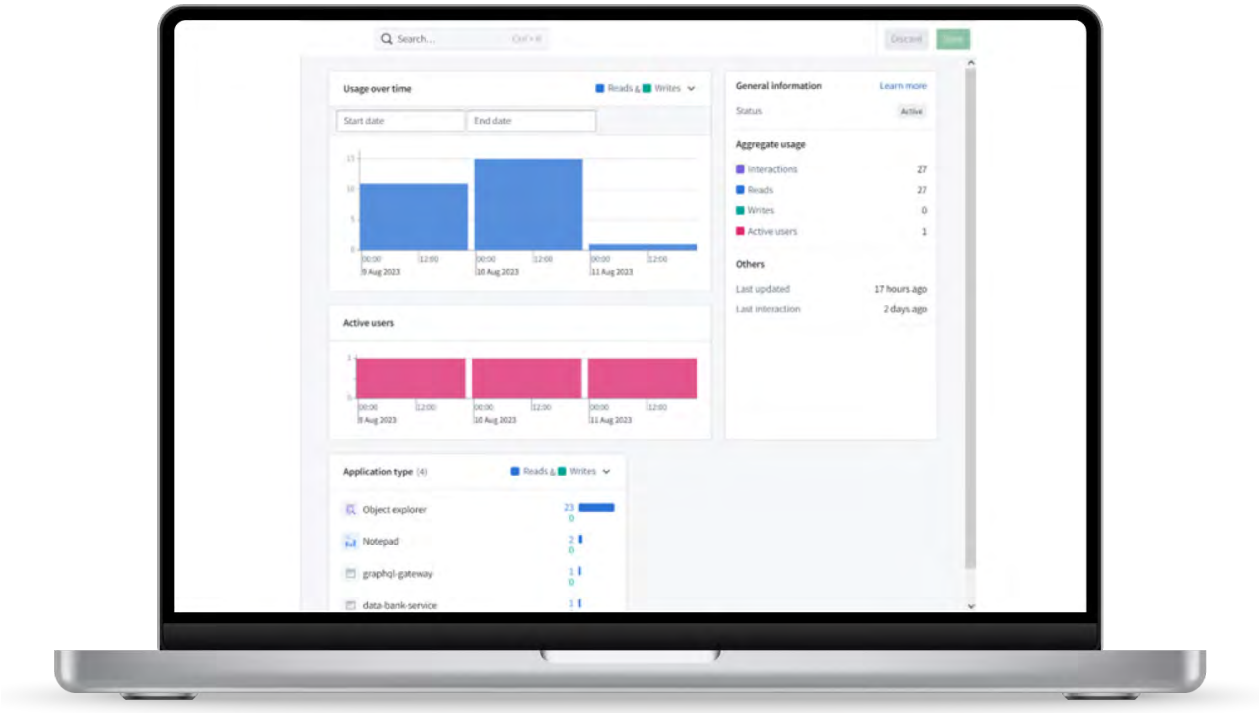


Figure 14: Object Usage Monitoring

[53] Palantir Technologies (2023) - Ontology - Foundry Rules
[54] OEDA shall support metrics regarding the data.
[55] OEDA shall support a mechanism to enable users to provide direct feedback to Data Providers.

8.3 Object View

The Ontology Manager is responsible for mapping fields, some of their behaviours and their semantic representation for an object type. The Object Explorer[56] tool is then used to view the actual objects:

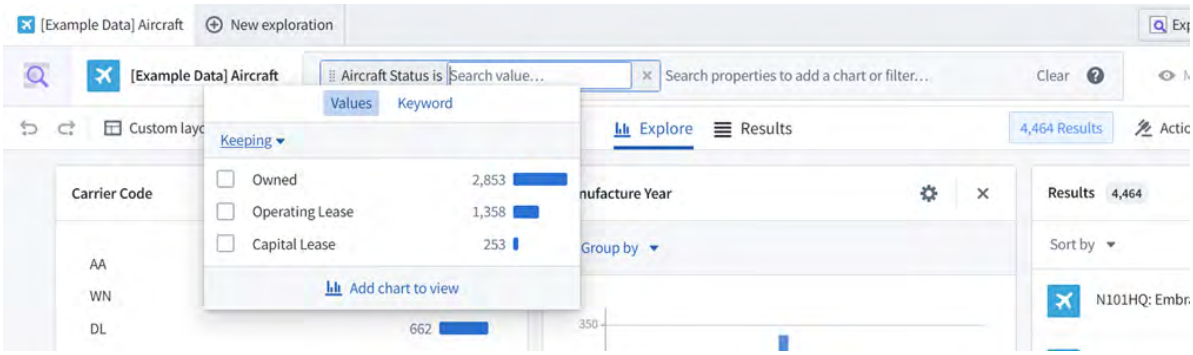


Figure 15: Filtering Data with Object Explorer

The tool provides a no-code interface to supporting explorations of the ontology through individual objects and their links. In the example above, the user is presented a profile of the Aircraft Status field to guide the exploration. Individual objects can be viewed by selecting the results tab and selecting a single instance:

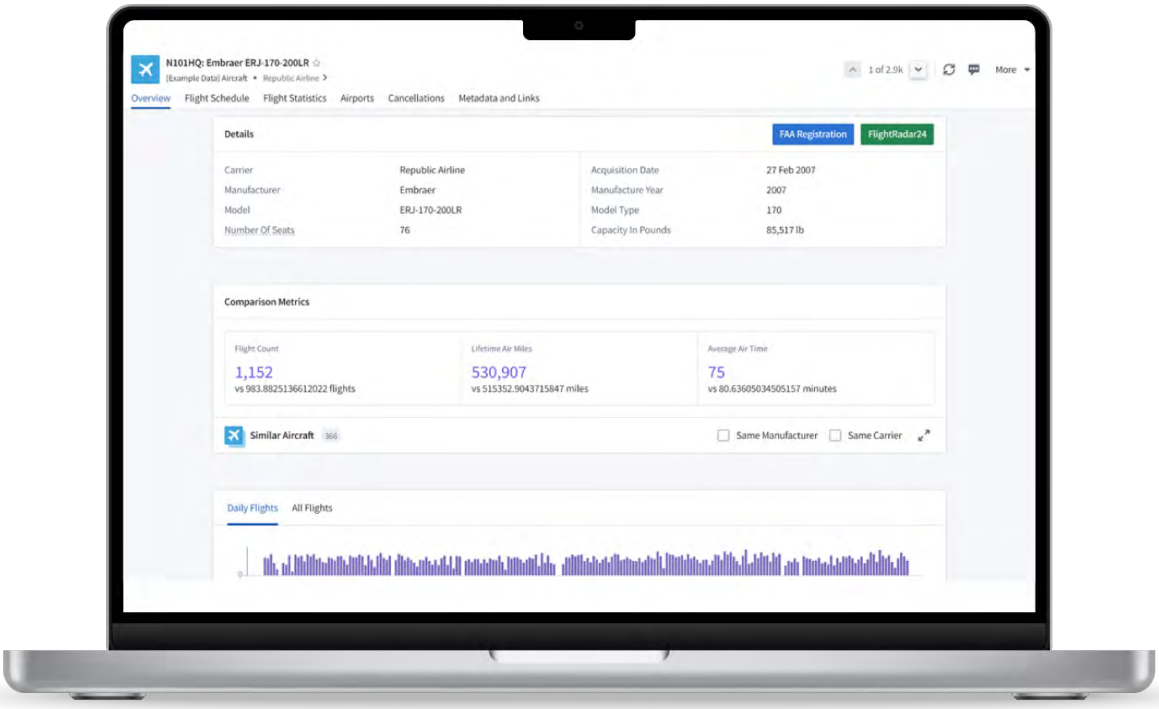


Figure 16: Example Object View

[56] Palantir Technologies (2023) - Ontology - Object Explorer

8.3 Object View

Note that the object view contains multiple tabs, has details presented as tables, comparisons with the rest of the population, as well as visualisations. There is a no-code approach to editing and customising these features to support the desired workflow and shows the native features of Foundry can offer significant visualisation and analysis capability based on the ontology. The language used to customise the object view is similar to desktop and web-based application development such as layouts, tabs and widgets. As each property has a semantic meaning, the customisation options presented are pre-filtered to ensure they make sense.

The following is a proposed mock-up for the OEDA Data Catalogue entry based on the available customization options[57]:

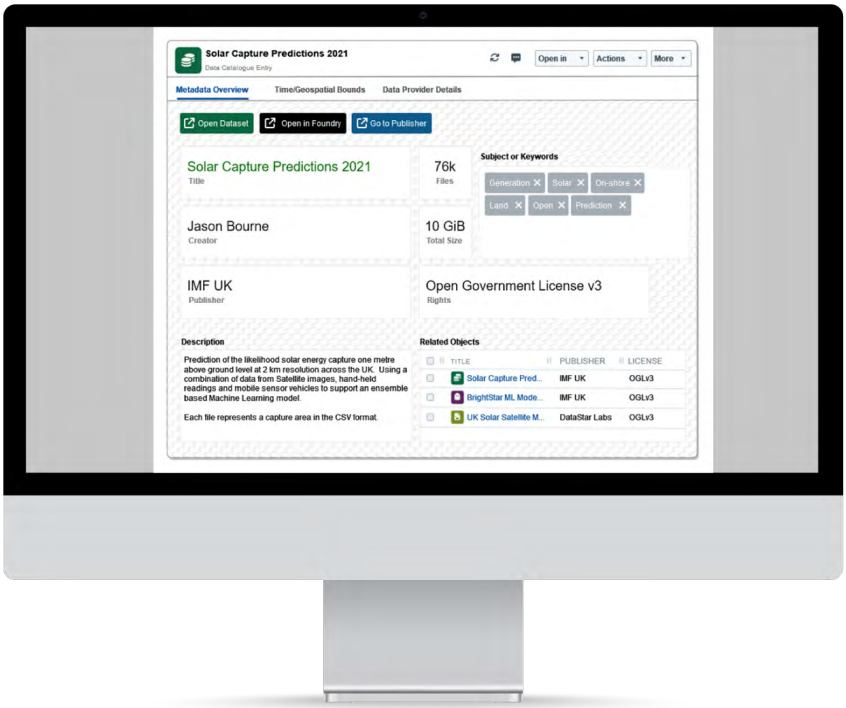


Figure 17: Proposed Object View Mock-up for OEDA

The mock-up object view shows an example Data Catalogue Entry Object Type, with three tabs; the first provides an overview of the entry in the catalogue. The second tab can include maps and/or timelines based on the coverage metadata - the intention is to identify other entries that are either within a set physical range or a time period. The third tab is intended to utilise the object link to the publisher or data provider to provide an alternative view to related catalogue entries.

The default tab shows three sets of buttons, which is (as all object properties are) backed by a dataset. The green button is based on the identifier element, which is likely to be a URL to the dataset hosted by the data provider and satisfies Requirement E5[58]. Where appropriate, a link to the same dataset within Foundry could be populated for the black button and satisfy the Requirement D5[59] for data exploration. The blue button demonstrates that other links could be provided to satisfy other requirements such as direct access to the external data provider host (e.g. web page) or an internal data provider object.

Four large property cards reflect various metadata elements and two smaller cards show attributes specific to this type of entry - the number of files Property for example is not appropriate for a REST API endpoint. The subject or keywords are presented as a filter, which controls the list of related objects in the bottom right of the figure. This contains multiple object types and not just restricted to the one data catalogue entry type. The description field is presented as a long text widget, which can support rich formatting using the Markdown language satisfying Requirement D4[60].

The mock-up illustrates a potential improvement in the ontology design that reflects the discussion around metadata standards captured in the Data Sharing Landscape report. Given the variety of data expected in the offshore sector captured within the OEDS report, discussions around minimum metadata requirements are reduced to the lowest common denominator, thus reducing the available context (and potential utility) for all users. The ontology offers a mechanism to capture the relevant metadata for all parties in a manner that provides the required contextual metadata but still meets the minimum requirements.

[57] Palantir Technologies (2023) - Object View - Configuration
[58] OEDA shall support external URL redirects, HTTP based APIs, the means to redirect to static files and other protocols to support streaming applications.
[59] OEDA shall support the exploration of data with either internal or external platforms.
[60] OEDA shall support rich formatting of content.

8.3 Object View

The current ontology design in effect takes a tabular view of the output and splits the object types by column. To enable context specific object properties, the catalogue could be split by row based on the Type element. The most practical implementation within Foundry is to automatically split the backing dataset into multiple tables with common columns to reflect the common object properties and additional columns that are relevant to that that type of entry to enable custom object views:

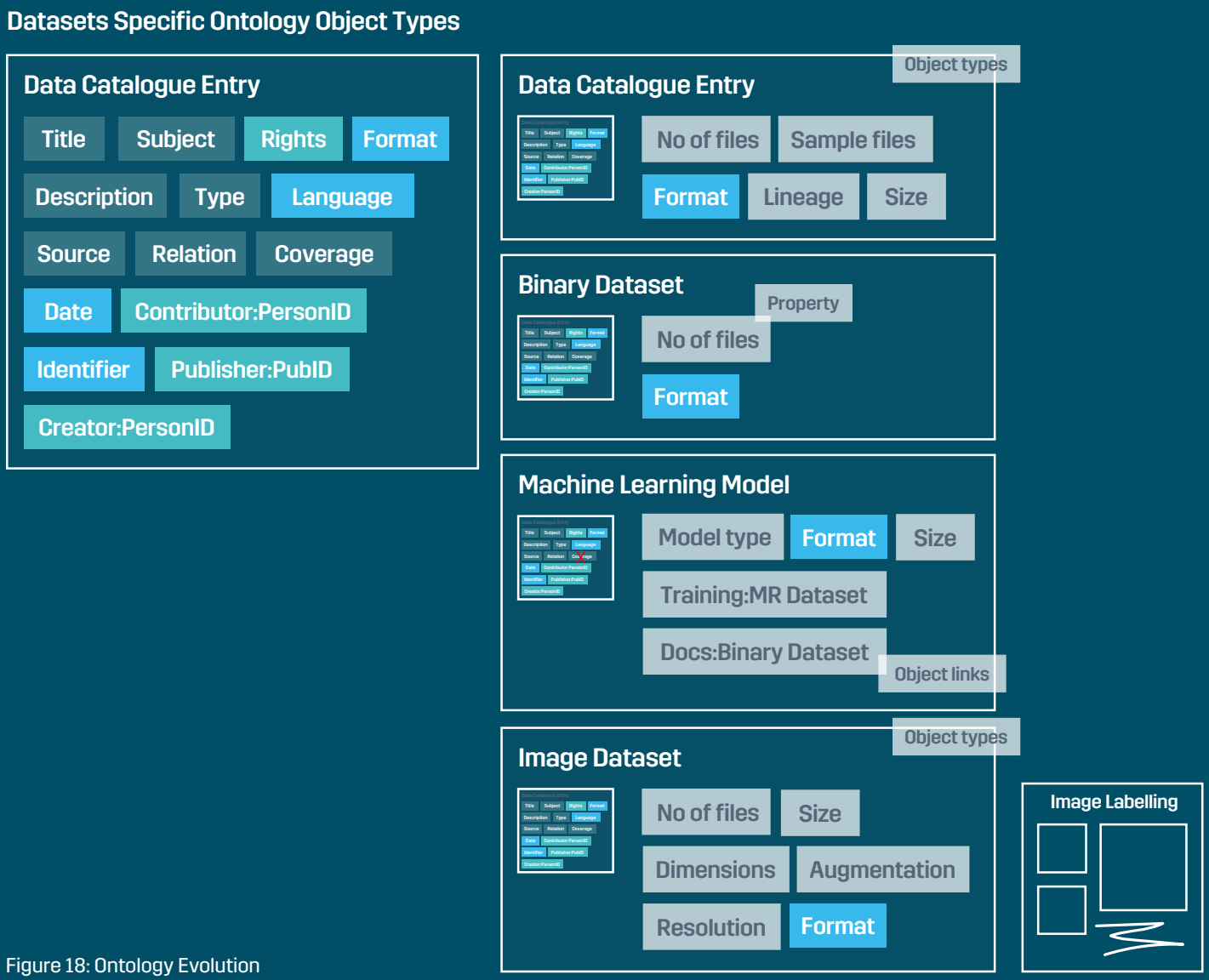


Figure 18: Ontology Evolution

8.3 Object View

To better accommodate the wider range of data sets expected within the offshore sector, the example above shows four data catalogue entry variations that empower data providers and the wider industry to provide the most appropriate context. The first example is of a machine readable dataset, where it is important to understand the number of files, their total size, lineage and attachments to samples. For a binary dataset (e.g. a PDF report), these properties may not be appropriate or needed. For a Machine Learning Model, the No of Files Property is not as significant as Model Type and it may have object links to the training data and to supporting documentation. The final example illustrates properties that may be useful for an image dataset.

Having a dedicated object type enables the designer to create and pin appropriate applications suitable in that context through the applications Sidebar[61]. In the example above, an image labelling application is created to capture labels to support a computer vision or object detection Machine Learning model project. This concept can be extended to meet other OEDA Requirements that involve a ticketing system or feedback capture workflow by creating a backing dataset to capture comments or the status of an entry, generating a suitable object type and linking them to other objects.

As this is a common design pattern with most Enterprise environments, the Foundry Training[62] and Documentation provides tutorials based on an aviation example to replicate the behaviour discussed in this report. The Ontology and the associated native features of Foundry will therefore satisfy OEDA and Data Practitioner Requirements E7[63], E8[64], D3[65] and D6[66].

[61] Palantir Technologies (2023) - Object Views - Applications Sidebar
[62] Palantir Technologies (2023) - Palantir Learning
[63] OEDA shall support means for prioritising data sets, either for release, update or additional context.
[64] OEDA shall support a mechanism to enable users to provide direct feedback to Data Providers.
[65] OEDA shall support data profiling for machine readable formats and support the hosting of sample data for user preview.
[66] Data Industry expectations for data format, structure and size are required prior to previewing the data - particularly important for larger datasets.

9.0

Embedding Enterprise Workflows

Once the ontology has been constructed, it can be explored with Foundry native features such as Object Explorer, Vertex for constructing and evaluating a system level Digital Twin and Map for geospatial based exploration. As stated for the pilot architecture, it is recognised that using Foundry native features requires some degree of training on the platform. To support a range of users, Foundry offers an application building service based on three components.

Workshop[67] enables the creation of high-quality desktop and mobile applications, with a range of no-code, low-code and code-based widgets utilising the ontology:

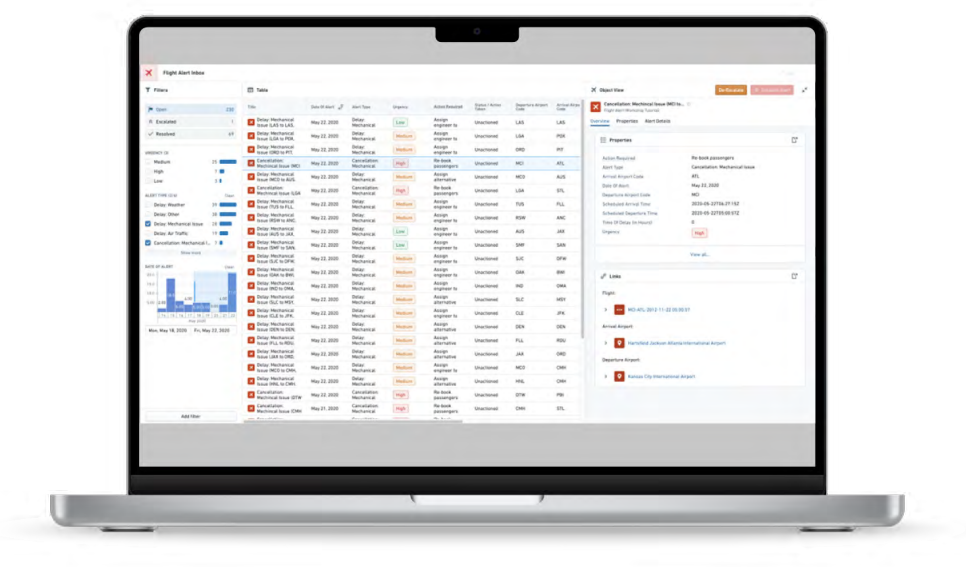


Figure 19: Example Workshop Application (from Palantir Documentation)

The example shows a Flight Alert Inbox application, where there are predefined filters in the top left, some high-level metrics in the left side bar with a histogram that can be used to further filter the data. The relevant objects or in this case, flight alerts, that match the filters displayed in the centre and a preview pane for a given object on the right-hand side. There are two prominent actions in the top right-hand side for this application.

Although this is an aviation example, the components on display could easily be transferred for any use case that requires triage such as Requirement E7 to support data providers prioritise data sets. A similar workflow could be used to identify objects or data catalogue entries that do meet a metadata standard or assess the impact of removing a dataset.

For reporting through Dashboard like Applications, Foundry offers Slate:

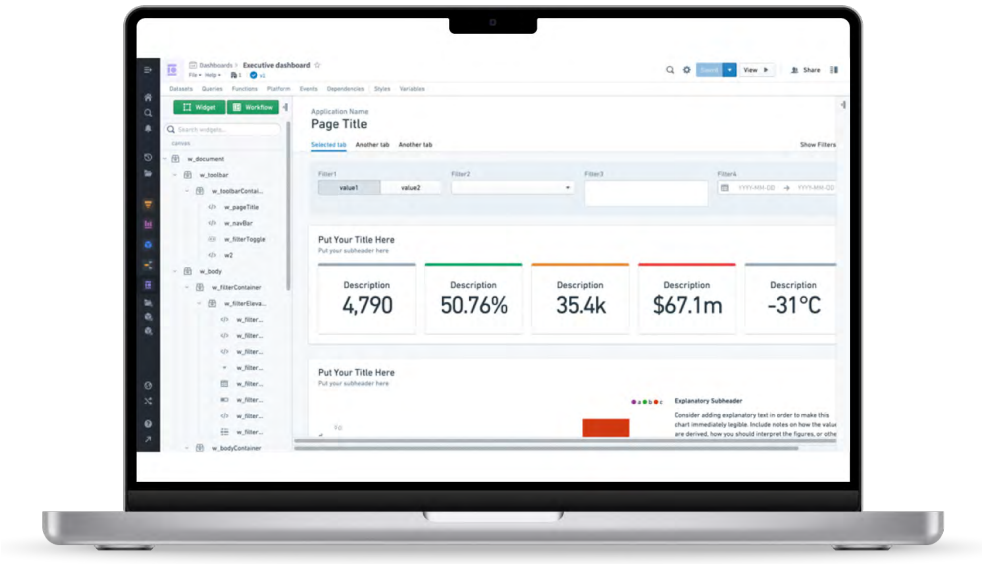


Figure 20: Example Slate Application (from Palantir Documentation)

Palantir Technologies (2023) - Application Building - Workshop

Although Slate has a drag-and-drop interface, customisation requires knowledge of web technologies and as such, is more suitable for builders who are familiar with that type of development workflow. The example above could be used to inform the community and stakeholders basic metrics on the performance of the catalogue. A single template to support data provider metrics could be generated and shared industry wide.

Given two apparently overlapping capabilities with Workshop and Slate, the differences from a user perspective can be minimal. For the builder, Workshop provides a desktop and mobile application development experience and Slate a web application experience. Typically, transactional use cases may be better served with Workshop, whereas high level metrics and exploration with occasional actions may be better suited with Slate. Although both use case types can be replicated in both development environments.

For a user, once an application has been created it in effect enters the Application Portal, which is equivalent to the Programs or Start Menu on most desktop operating systems. For a sector wide implementation, there is likely to be 10s or 100s of applications and therefore to simplify the user experience Foundry offers a method to customise the overall experience. For example, a data user will not need access to the same applications as a data provider. To group or integrate applications together, Foundry offers Carbon[68]:

Based on the user's profile, it is possible to offer a custom Start Screen that displays an integrated portal that combines Foundry native features (such as Object Explorer, Contour for analysis etc) with custom applications (from Workshop and / or Slate). The example above provides the key features for any modern operational platform. The header not only contains tabs but integrates notifications from Foundry and various applications.

Despite the level of customisations and the builder experience available to create Enterprise applications, if that is insufficient or external access is required then the Foundry API[69] is available that provides access to the ontology but also the related actions. This enables the builder to generate very similar applications outside of Foundry if required. As stated for the pilot architecture, this feature is required to provide users access to the catalogue for open datasets without authentication.

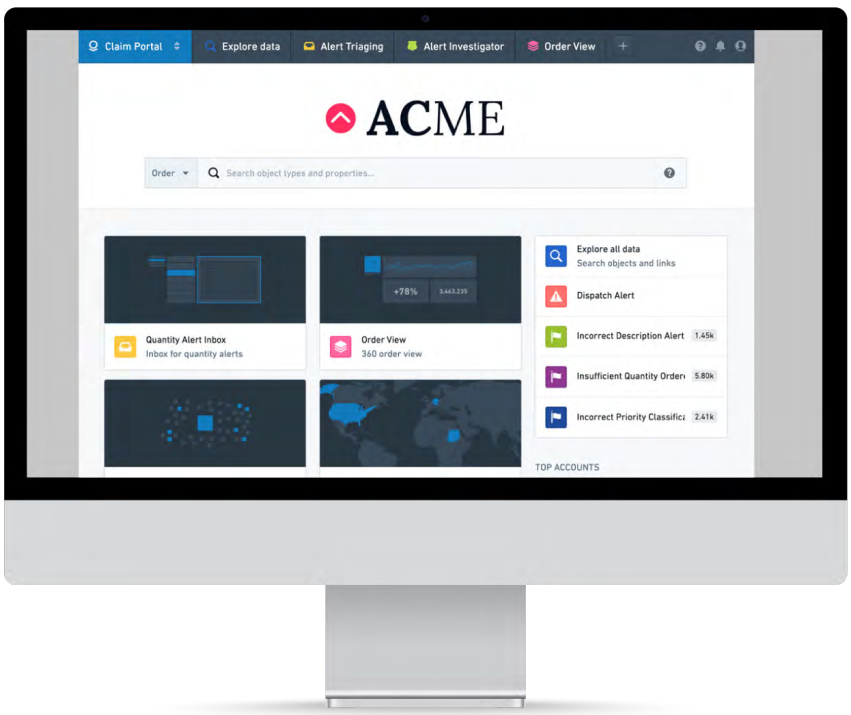


Figure 21: Example Carbon Application (from Palantir Documentation)

Palantir Technologies (2023) - Foundry API

10.0

Conclusion

It has been demonstrated that all of the OEDA and data practitioner requirements with the exception of E3 - the platform basis should be open-source software - can be met with Foundry. Furthermore, features such as the ontology can be used to construct multiple approaches in creating a data catalogue from the same backing dataset but with different object types. Once an ontology has been defined, constructing operational workflows in an agile manner is supported through Foundry native features or using custom applications.

It's proposed for the pilot that three potential interfaces to the catalogue are provided to demonstrate Foundry's suitability for OEDA, using object viewer, a carbon application and an externally hosted web application to satisfy the requirement to provide access without authentication.

11.0

Appendix A: OEDA Requirements

The two tables are an extract from OEDA Report 1 - Data Sharing Landscape:

Req. ID	Requirement	Source(s)	Compliance Statement
E1	OEDA shall support the OEDS defined data catalogue.	From Action 2.1: Offshore Energy Data Catalogue (OEDC).	Catalogue and metadata aggregator demonstrated through the Pilot Architecture (p13) and Data Integration (p15) sections.
E2	OEDA shall support the OEDS defined Data Sharing Fabric.	From Action 2.2: Data Sharing Fabric (DSF).	Foundry's security features can satisfy this requirement through SSO and Data Governance (p17).
E3	OEDA shall be based on open-source software and open standards. It should facilitate the Presumed Open principle.	The principle of being as "Open as possible" as expanded in the EDiT report as: "Wherever possible, it is proposed that these should be based on open source software, open data licences and open standards"	This requirement is not met as Foundry is not Open Source software as discussed in the Pilot Architecture (p13) section.
E4	OEDA shall support a customisable set of attributes to act as metadata and have the means to define differing levels of priorities and controls.	Several metadata attributes have been defined, in effect the superset from Ice Breaker One on Open Net Zero , EDVP and Dublin Core but recognising the need to set and control differing priorities.	The Ontology Design (p22) and its Backing Dataset provides the flexibility to meet this requirement.
E5	OEDA shall support external URL redirects, HTTP based APIs, the means to redirect to static files and other protocols to support streaming applications.	The ONS Energy Data Visibility project stated the protocols initially should be HTTP based, but recognised with maturity it should support streaming applications.	The URL to a source can be hosted as a button or link in the Object View (p25).
E6	OEDA shall support metrics regarding the data.	The EDVP identified the need to surface and measure data quality - the subjective component in assessing data quality will be influenced by existing Industry standards-based initiatives. The implication is users manually submitting feedback.	Requirement is met with multiple features: Data Pipelines and Data Lineage (p18), Object usage in the Ontology Design (p22)
E7	OEDA shall support means for prioritising data sets, either for release, update or additional context.	Multiple reports including EDVP and EDTF cited a two-phase approach to data sharing, where users can see a list of potential sources and request them. These are then prioritised for release based on requests received.	Foundry can support a ticketing-like application that would enable the prioritisation of data based on feedback through the Object View (p25) and Embedding Enterprise Workflows (p29).
E8	OEDA shall support a mechanism to enable users to provide direct feedback to data providers.	Multiple reports have cited providing feedback between users and data providers, the former to help improve the data sources and the latter to support internal business cases.	Requirement is met with multiple features: Data Pipelines and Data Lineage (p18), Object monitoring (p22), a ticketing system through the Object View (p25).
E9	OEDA shall display lineage or provide the means to define a lineage between datasets. OEDA shall support datasets to be related using attributes.	EDVP also identified the need to establish both data provider led and user driven relationship mapping between datasets.	Full access to the Data Pipelines and Data Lineage (p18) provides the visibility even across organisational boundaries.
E10	OEDA shall support and maintain support for the highest security standards in the field of Authentication, Authorisation and Zero Trust (including defence in depth).	OEDS report states in Action 3.2 Cyber Security: "The offshore energy sector should continue to prioritise cyber security, adhering to cyber security best practice and disseminate progress to the wider sector to help developing industries."	The Security layer in the Foundry Introduction (p9) demonstrates the highest cybersecurity standards.

Table A1: Technical Requirements derived from the Energy Sector

[70] Energy Systems Catapult (2022) - Delivering a Digitalised Energy System
[71] Icebreaker One & Open Net Zero (2023) - Open Net Zero by Icebreaker One
[72] Hippo Digital (2020) - Energy Data Visibility [Discovery report]
[73] Dublin Core Metadata Initiative (2023) - DublinCore

Req. ID	Requirement	Source(s)	Compliance Evidence
D1	OEDA shall support the use of internal and external repositories for dataset documentation, context, data samples, API definitions and other assets.	Data Industry expectations around open source software development and documentation culture.	Requirement is met with multiple features: Data Pipelines and Data Lineage (p18) illustrates workflows, whilst Data Integration can ingest from external repositories.
D2	OEDA shall support the use of long held security tokens including but not limited to client and server-side certificates - mutual Transport Layer Security (mTLS) with Hardware Security Modules (HSM) and/or rotated authentication tokens (i.e., OAuth 2.0 / OIDC).	Recommendations from wider energy sector reports are tilted towards Human interaction. The OEDS report explicitly states the use of machine-to-machine interactions. The data industry expects the use of standard protocols and approaches.	Demonstrated through the use of an OAuth2 client in External Catalogue Access (p14).
D3	OEDA shall support data profiling for machine readable formats and support the hosting of sample data for user preview.	Data industry expectations for data format, structure and size are required prior to previewing the data - particularly important for larger datasets.	Through the Object View (p25), a profile of the data on the Ontology can be established or through custom Object Types.
D4	OEDA shall support rich formatting of content.	The open-source development culture also provides rich documentation around a project that users can collaborate on, which can also be hosted externally.	Widgets in the Object View (p25) illustrate support for rich formatting through Markdown.
D5	OEDA shall support the exploration of data with either internal or external platforms.	Kaggle has demonstrated that users prefer to make their own assessments of the data rather than rely on data provider attributes. This includes the principle of the data being Open to Explore, either externally much like the Python Data ecosystem with Binder or internally through hosted Jupyter computational notebooks.	Exploration of the data within the platform and through external links is demonstrated through Object Views (p25).
D6	OEDA shall have the means to support a variety of metadata formats (beyond the current attribute-oriented needs).	Data Industry expectations for data format, structure and size are required prior to previewing the data - particularly important for larger datasets.	The creation of bespoke Object Types in the Ontology Design can cater for different metadata formats as well as choice of a Backing Dataset (p25& p28)

Table A2: Proposed Requirements from the Data Industry

Net Zero Technology Centre

Technology Driving Transition

Contact number:
+44 (0)1224 063200

Media enquiries:
pressoffice@netzerotc.com

Net Zero Technology Centre
20 Queens Road, Aberdeen AB15 4ZT

www.netzerotc.com