e-Fish Data Architecture

FRDC

Submitted by

York Stanham
Business Analyst
M. 0422 826 050
E. york.stanham@mtpservices.com.au

04 March 2020

mtpservices.com.au

Contents

Document control

Version	Date	Changes made	Author
0.1	17/09/2019	Initial draft	York Stanham
0.2	26/09/2019	First review changes made. Included more examples	York Stanham
0.3	09/10/2019	Final changes after AFMA review	York Stanham
0.4	12/10/2019	Addition of section 6	Andres Pinzon York Stanham
0.5	31/10/2019	Addition of figure 8 and clarification of some points	Andres Pinzon York Stanham
0.6	07/11/2019	Added place holder headings for additional information. Updated Figure 8	York Stanham
0.7	04/12/2019	Added workings of PoC to the document	Peter Monus Vebushan Rajendran Andres Pinzon
0.8	21/01/2020	Version after feedback from AFMA	York Stanham
1.0	04/03/2020	Final version following internal review	AFMA

1. Executive summary

This report provides an analysis of a service orientated architecture in the context of a fisheries management agency's ability to collect, share, and process data. The report includes examples from a proof of concept application. This application was developed through consultation with AFMA staff and aims to illustrate how a system might meet the design principles uncovered in the first stage of the e-Fish project. The five design principles being:

1. **Linked data** – Data sets are inherently linked in a way that allows ease of use.
2. **Modern data sharing** – Data sets should be exposed to external users through an easy to maintain and minimal touch solution such as application programming interfaces (APIs).
3. **Ensure data integrity** – Data is clean and validated with minimal errors. Data is stored according to predefined elements maintained in an agency or industry wide taxonomy.
4. **Standardised data collection** – Data is received in a uniform approach. Care is taken to not duplicate data where it is unnecessary to do so.
5. **System capability fit for purpose** – Implemented systems directly support various business outcomes of fisheries stakeholders.

Service-oriented architecture is a style of architecture that makes use of individual services. In the case of a fisheries management agency, these individual services are essentially units of logic aimed at performing one business function, such as (but not limited to):

- A process that performs quota decrements
- Authentication of a client's details
- A report that returns active fishing vessels

Advantages of using a service-oriented architecture to business processes include:

- Collaboration
- Reuse
- Increased availability
- Reliability

The proof of concept application discussed in this report makes use of a service-oriented architecture built using Microsoft azure services. This proof of concept application uses a Customer Relationship Management (CRM) system and single page applications to link and share data often seen in a fisheries management system, this includes vessel details, licencing details, and concession holder details. The proof of concept has also included the linking and comparison of electronic monitoring data and daily fishing logs. This comparison has allowed for various metrics including the reporting accuracy for each vessel and how this compares to the fleet average.

This report also highlights the lessons learned during the development of the proof of concept. These lessons including further understanding of the functionality and usefulness of a dedicated CRM system within the context of a fisheries management agency. During the development of the POC, issues around the way information is currently being collected were also identified. To allow for data to be linked in the future, the project showed a need to standardise the collection process and seek to put in place controls

during the submission process to insure enough information is collected to relate it to existing data. Basic validations such as checking the data relates to known clients and vessels should be implemented during the submission process as post hoc identification based on other captured information was often not sufficient.

2. Introduction

The e-Fish project is investigating a solution for how a fisheries management agency could improve the connectivity between data capture systems and allow insight into the interactions clients have with the agency.

Fisheries management agencies provide a range of services to their clients to pursue their legislative objectives. To support agency operations there is a need to be able to relate information received by an agency's systems and deliver a complete picture of client interactions. Due to the overlap between many fishery management agencies' services, a "tell us once" approach to information provision is also needed with changes made to one system reflected across the other interactions their clients have.

The e-Fish project is sponsored by the Fisheries Research and Development Corporation (FRDC). The project's outputs are applicable to any fisheries management agency in Australia, noting that individual agencies will have their own varying systems so the architecture will need to be customisable to account for this variability.

2.1 Scope and Purpose

The scope of this document is to provide an overview of the recommended architecture for a fisheries management agency and management system for fisheries data. It builds on the Design principles and consultation for the project as described in "e-fish: Design principles" document. This document captures high-level system designs, database setups, and both internal and external data sharing mechanisms.

The purpose of this document is to demonstrate how the proposed data architecture applies the design principles listed in Section 4. The document describes how the proposed data architecture applies each principle using real world examples where relevant.

3. Data architecture design

A fisheries management agency collects, analyses and shares data in an effort to maintain fisheries. This data often includes information around what activities are taking place in fisheries, what fish are caught, retained, and thrown back, and any wildlife interactions that may occur. Additional data also collected by fishers include position data of vessels, and landing reports by fishers and fish receivers. Data is used in a number of ways including tracing fish from the ocean to the plate, conducting stock assessments, and ongoing management of protected species. Along with collected data, fisheries management agencies also maintain client data including concession holders, boats and authorities.

A fisheries management agency uses information collected, stored, and processed through the combination of individual, but tightly coupled, services. Each service is designed to meet business needs which are often driven by legislation or research outcomes. Such services generally include:

- Daily fishing logs
- Landing data
- Client data including licencing, boats, and authorities
- Observer trip records
- Vessel monitoring system (VMS) data
- Electronic monitoring system data

Due to the modular nature of the business processes in place, a modular architecture is recommended. This modular architecture is known as service-oriented architecture (SOA).

3.1 Service-oriented architecture (SOA)

Service-oriented architecture is a style of architecture that makes use of individual services. In the case of a fisheries management agency, these individual services are essentially units of logic aimed at performing one business function, such as (but not limited to):

- A process that performs quota decrements
- Authentication of a client's details
- A report that returns active fishing vessels

A highly simplified service could look something like Figure 1: SOA Example. This figure shows a simple breakdown of fishery agency services into isolated components linked to their own databases. The blue arrows are linked between the services; these could be RESTful APIs or perhaps a service bus. Green arrows are the links between data collection services and their respective databases. The black arrows are data flowing out of the agency. This figure also illustrates the types of services that can be placed within the system; this includes things like the compliance portal, or notifications to mobile phones, and also externally exposed APIs exposing aggregation of data from numerous services.

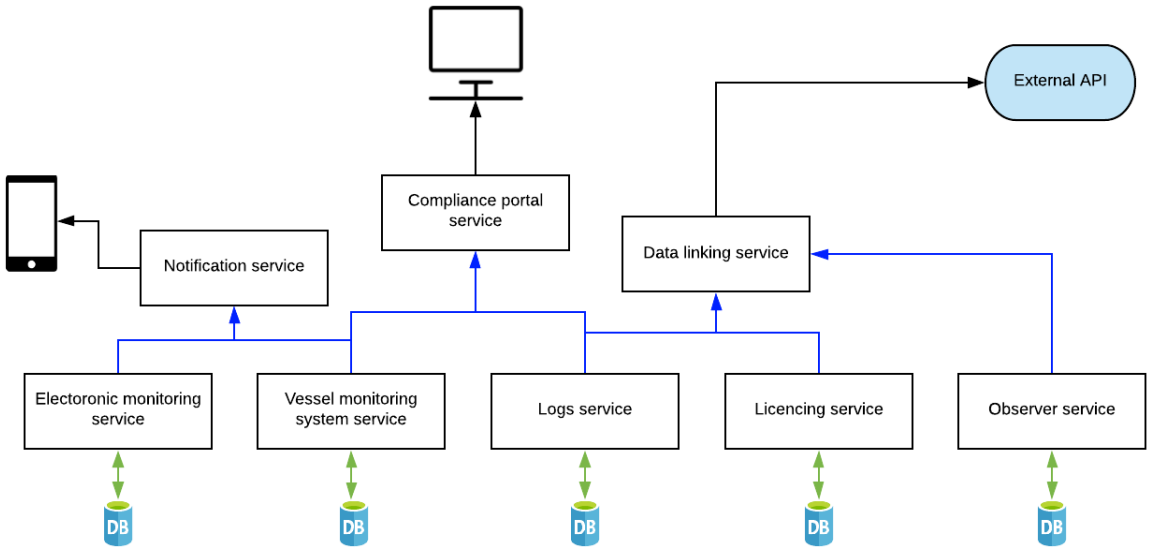


FIGURE 1: SOA EXAMPLE

An individual service will have the following characteristics:

- It can access other services within the same ecosystem. By using triggers within functions, http requests and services buses, a service can respond to requests or events from other services in order to facilitate a business function. For example, the quota decrement service can be triggered when an accurate weight is received from a landing report.
- It is loosely coupled with the rest of the ecosystem. This allows relatively easy updates to a service without the need to retest the entire enterprise solution. This also facilitates the sharing of modular services between fisheries management agencies. Services could be duplicated and shared, allowing individual governance, or a shared service accessible by numerous agencies could be developed.
- It can be exposed externally to allow interactions between a fisheries management agency and external users such as fishers, CSIRO, ABARES, or data users from other fisheries management agencies.

SOA focuses on business processes, modularity and reuse. The implementation of SOA to a new or existing system is often considered in the following cases:

- Existing software applications are generally large and encompass all aspects of business uses such as data collection, sharing, manipulation, and user interface components leading to extensive testing or large down times to upgrade small pieces of functionality.
- Existing software applications are developed in a somewhat unstructured manner, often having functionality layered on top of existing functionality, due to the evolving nature of business requirements.
- Large system-wide software applications are hard to understand from a business perspective because upgrades are often performed in the form of patching existing applications rather than redeveloping additional standalone services as new requirements emerge.

These points are illustrated in Figure 2: . In this example, traditional monolithic systems are shown on the left and a SOA architecture is shown on the right. On the left the systems and business functions all contain their own logic, often duplicated in other services, whereas the SOA on the right has defined business functions that call the isolated services if needed.

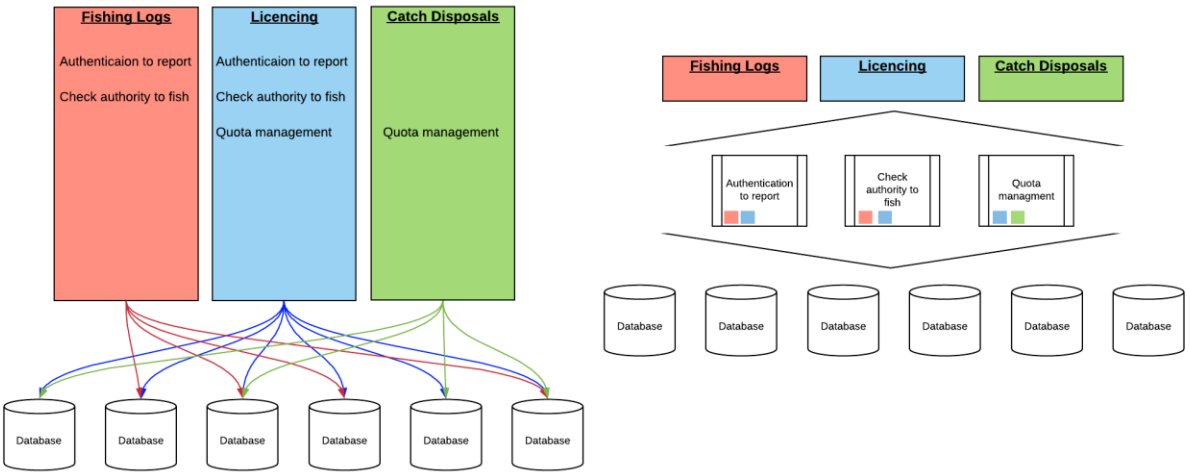


FIGURE 2: MONOLITHIC APPLICATIONS VS SOA

3.1.1 Advantages to business processes

Implementation of a SOA provides several advantages to business processes at a fisheries management agency, largely due to the modular nature of the services. As opposed to a siloed system architecture where there is little communication between services yet large dependencies, the services within a SOA can be developed independently and can have fewer dependencies.

3.1.1.1 Collaboration

The modular nature of services within a SOA allow for better understanding of the purpose of each service, even if software knowledge is limited.

This is especially useful when business users are considering updates or changes to services. When using a SOA, the implications of such changes can be easily understood and therefore facilitates the inclusion of business users in the design, scope, and purpose of each service.

This is likely to expedite the changes needed when different business outcomes arise, such as collecting new data from fishers, or sharing additional data with an external agency.

3.1.1.2 Reuse

Sufficiently modular and well-designed services lend themselves well to reuse, either within or across fisheries management agencies.

The ability to reuse services benefits business when upgrading existing functionality or implementing new functionality because it allows easy retainment of services not undergoing a change. The reuse of

existing services in a SOA also largely reduces the testing effort required when compared to redeveloping an entire system from scratch.

3.1.1.3 Increases in the availability of data

SOA is often used to increase the availability of data to both internal services and to external users. The increase of data availability within services is a key component of any SOA design. The data is passed around between services through the use of a service bus or API calls to be used in whichever service needs it.

External data sharing is primarily achieved through the development of services specifically aimed at exposing data to approved external users. These data are then exposed through an API. The granularity of data shared externally can be driven by utilising different APIs per data set and also through employing logic around the data each user has access to.

3.1.1.4 Reliability

Services that form part of a SOA are often more reliable than monolithic enterprise solutions. This is due to their smaller size and loose coupling with the rest of the architecture which makes testing efforts much easier and more isolated.

If changes or updates are made to a particular service, for example collecting an additional data field from fishers, then only the affected service will need to undergo a full regression test, rather than the entire enterprise solution.

4. Design Principles and SOA

This section explores how the Design Principles can be met using a SOA. The Design Principles are:

1. **Linked data** – Data sets are inherently linked in a way that allows ease of use.
2. **Modern data sharing** – Data sets should be exposed to external users through an easy to maintain and minimal touch solution such as application programming interfaces (APIs).
3. **Ensure data integrity** – Data is clean and validated with minimal errors. Data is stored according to predefined elements maintained in an agency or industry wide taxonomy.
4. **Standardised data collection** – Data is received in a uniform approach. Care is taken to not duplicate data where it is unnecessary to do so.
5. **System capability fit for purpose** – Implemented systems directly support various business outcomes of fisheries stakeholders.

4.1 Linked data

During the first stage of the e-Fish project, we found that data is often stored in "silos" in many different formats and in numerous different locations. Internal and external data users are unsure where to find information and have low confidence in the reliability of information even if it is found.

Data available to internal users, data managers, and fishery managers is often duplicated and unable to be used until manually linked, often in spreadsheets. In some instances, the manual linking is limited to guess work based on dates, times, and GIS data rather than a common unique identifier.

Based on the findings from stakeholder interviews, a number of key challenges emerged:

- Reliably linking data sets that aim to capture the same or related data is often done based on a qualitative assessment. For example, linking electronic monitoring and logbooks, or linking VMS data with reported daily fishing log activities.
- Manual linking of data from disparate data sources is a time consuming and expensive process that often constitutes a significant amount of the work undertaken by fisheries managers and scientists.
- Manual linking of data often requires some massaging of data to produce a holistic picture. This massaging of data means the single point of truth is lost and also means agencies holding the original data are unable to reliably produce the same results and reports as those produced by some data users.

4.1.1 Using a SOA to reliably and automatically link data sets

Currently, data sets are mostly linked through a manual process by the data user. Adoption of a SOA approach will allow a data linking service to be developed. This data linking service will be a self-contained service that aggregates data from one or more data collection services. The transfer of data from the data collection service(s) to the data linking service can be triggered by a range of actions such as a new data entry, a type of data entry, or even at a specific time of day. Data can be transferred through

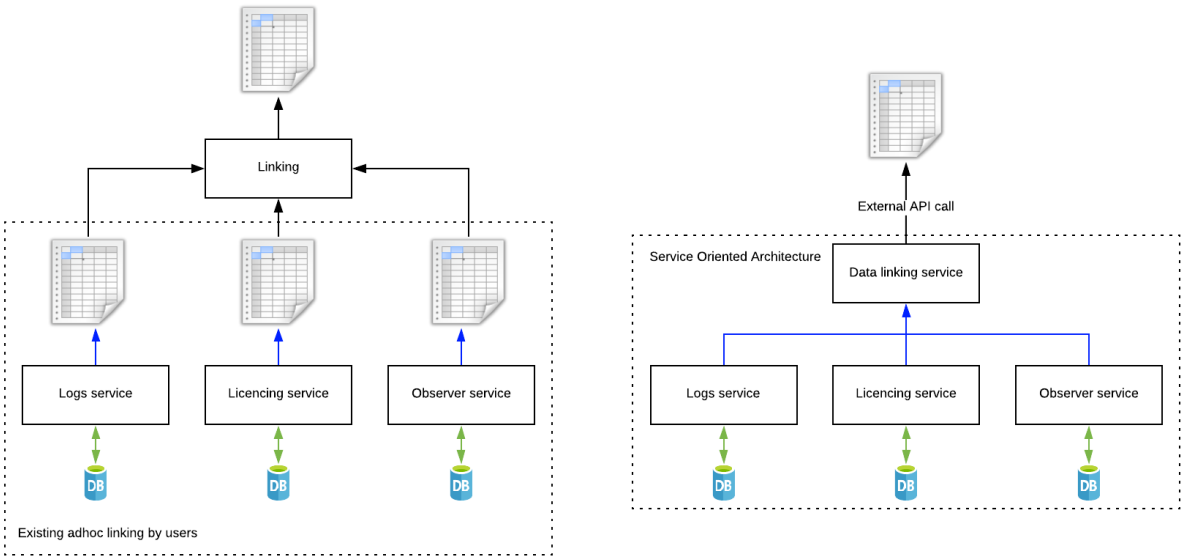
API calls between the data linking service and the data collection services or perhaps through the use of enterprise wide service bus.

This will allow end users to access data post linking rather than linking individual disparate datasets themselves. The utilisation of a linking service will ensure data is linked in the same manner every time regardless of which user is accesses, using, or processing the data.

How data is linked will be determined through the types of data that are being linked rather than through one unique identifier at appears across all data sources. For example, linking fishing trip information with catch disposals may be linked through a combination of the trip identifier and vessel identifier whereas linking the GPS tracking data with the fishing trip information may be done through the dates and vessel identifier rather than trip ID.

An example of a SOA using a data linking service is seen in Figure 3. Previously, users would access the data directly through a data-access service, then link the data manually, often using spreadsheets (left panel) with individuals determining the best or most accurate link. Because this data is linked by individual users on an ad hoc basis, the results and data collected is often not reproducible across users and time periods.

On the right, the user linking has been replaced by a data linking service. This linking service links the data in the same manner each time, ensuring the end product is the same irrespective of the user or the time period. The linked data is then exposed to end users via an API.



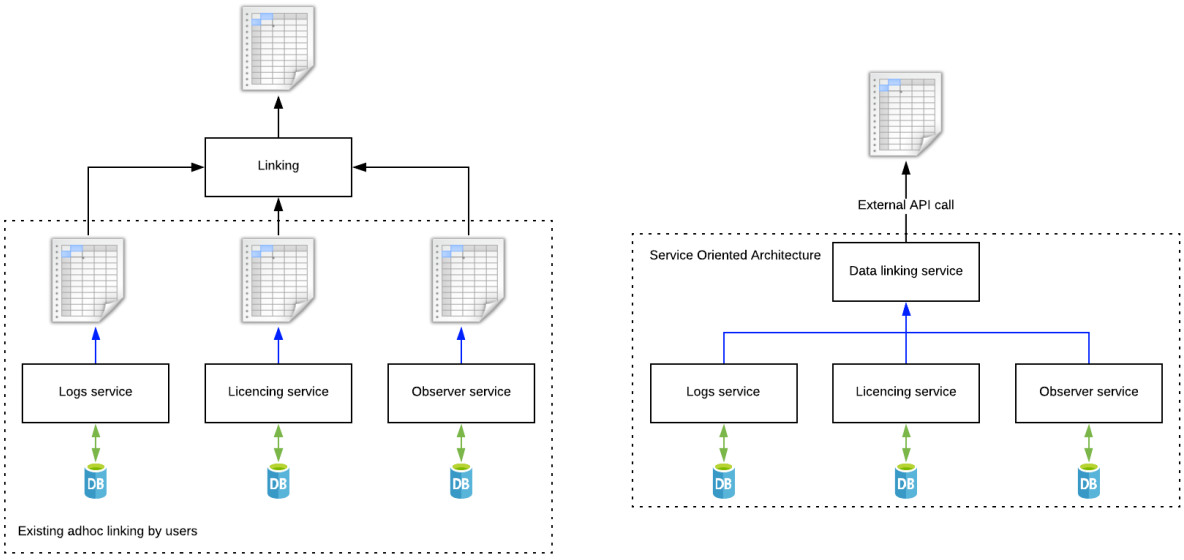
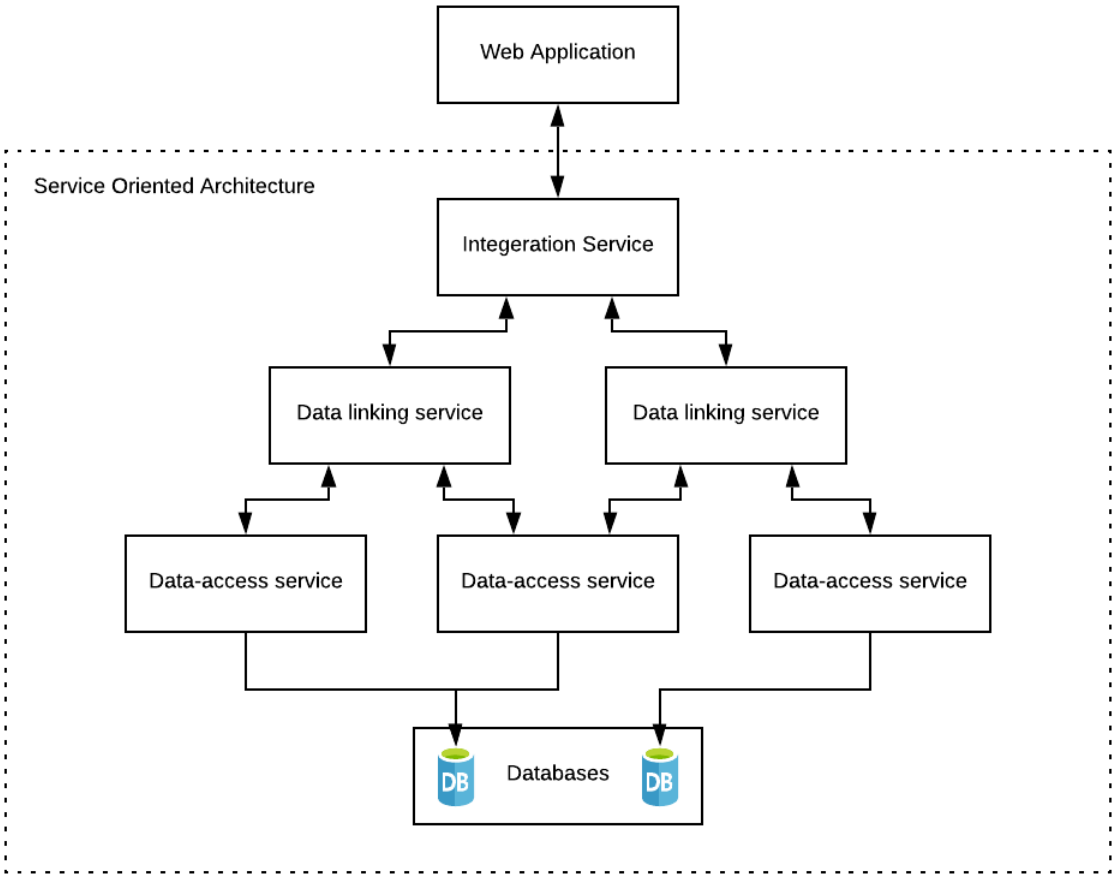


FIGURE 3: DATA LINKING SERVICE EXAMPLE



4.2 Modern data sharing

Fishery management data is shared frequently across and within agencies, scientific bodies, compliance and intelligence bodies.

Data is currently shared through the sending of spreadsheets (csv files) via emails, and copies of databases on DVD are mailed to the receiving party. The sharing of this data is often initiated through a request rather than a predefined agreement to send particular data at a particular frequency. There is no self-service option.

Based on the findings from stakeholder interviews, a number of key challenges emerged:

- Manually requesting and transmitting data often causes significant delays. There are often days between a request and receiving the data.
- Duplication of data to share with external agencies causes the loss of a single source of truth.
- Confidentiality and privacy obligations prohibits sharing of data across jurisdictions and to third parties.

4.2.1 Using a SOA to enable data self-service and retain a single source of truth

Services in a SOA communicate through messages. These messages could be service buses, SOAP/XML requests, or HTTP requests. Allowing external access to data can be achieved by exposing these internal communication methods externally or by developing additional externally facing API methods.

Data shared externally can be controlled through a number of mechanisms. Users accessing the APIs can be identified and their access verified before any data is exposed. The use of a verification service such as API gateway can also limit what data services, and therefore data, each user has access to. This might be useful when an agency wants to only expose certain data to a certain user. Adjusting what data or data fields are shown externally can be adjusted in the data sharing service rather than at the data collection service. This loose coupling of the data linking, and data collection service allows quick changes to data sharing as legislative barriers are removed or changed.

An example of internal and external data sharing is seen in Figure 4: Externally exposed Messaging. This figure shows a high-level setup of data linking services ultimately leading to a self-serve web portal. This figure also demonstrates how data from each linking service can be shared externally.

Within a SOA model, each service has total ownership of its data. Data is not shared with another service unless it is processed by logic within the service that owns that data. This means that the service that maintains

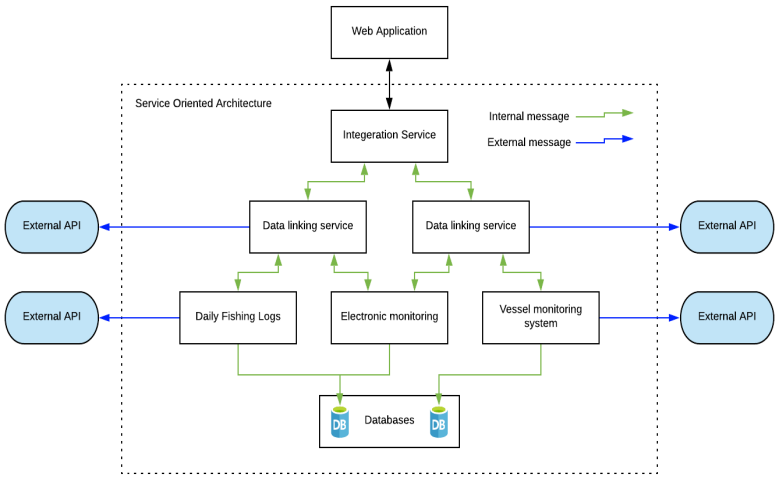


FIGURE 4: EXTERNALLY EXPOSED MESSAGING

ownership of a specific set of data always has access to the original data and any connected services make use of processed data.

4.3 Ensure data integrity

Data is often received by fishery management agencies in a way that suits the particular data collection stream (e.g. logbooks or VMS) and the integration method (APIs, paper based forms, spreadsheets etc.) used to deliver the data without considering the wider use of data collection and use within a fisheries agency.

Because varying data collection streams are associated with a particular fishing trip (e.g. logbook records, observer records, VMS) and current siloing of those streams, duplication of data occurs; often with different naming conventions, methods of collection, historical significance, validation and accuracy.

Based on the findings from stakeholder interviews, a number of key challenges emerged:

- The diversity of data collection methods and validation across systems leads to significant resources dedicated to cleaning and processing of the collated data before any meaning can be drawn.
- Users are often not aware of the accuracy of the data. This, in combination with data cleaning leads to insufficient evidence to make decisions.
- Siloed data from different, unique collection methods do not allow agencies to establish a single source of truth across all systems. This leads to difficulties and inaccuracies in any ensuring analysis of those datasets.

4.3.1 Using a SOA for single point of data collection and enabling data accuracy

Once developed, each data collection service can be exposed externally to users who report to a fisheries management agency. The data collected can then be exposed to a service that can accurately and reliably transform this data, so it is fit for purpose.

While using a SOA won't necessarily increase the accuracy of data collected, it will ensure valid data is submitted by using reference data and validation on fields through the development of dedicated, standalone services aimed at achieving this. It will allow data to be collected in a timely manner and ensure transformation of data is consistent, rather than relying on internal and third-party users to manipulate the data as needed. An example of such setup is seen in Figure 4: Externally exposed Messaging and in Figure 5: DATA Collection and processing services. Noting the example in Figure 5 can be adjusted to have data processing services acting over more than one data set.

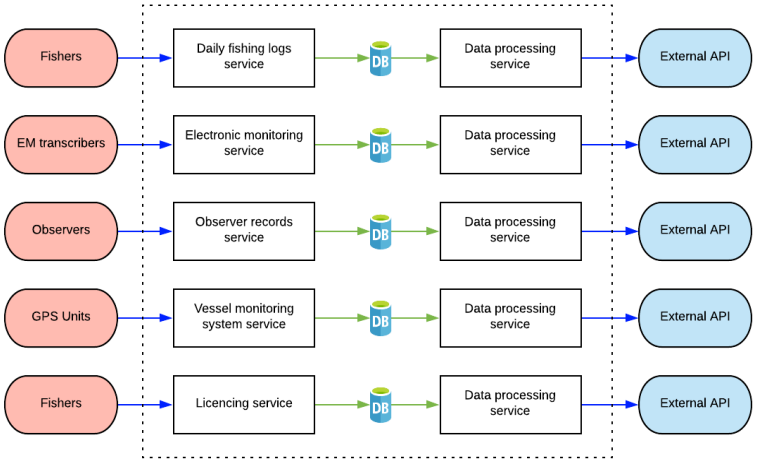
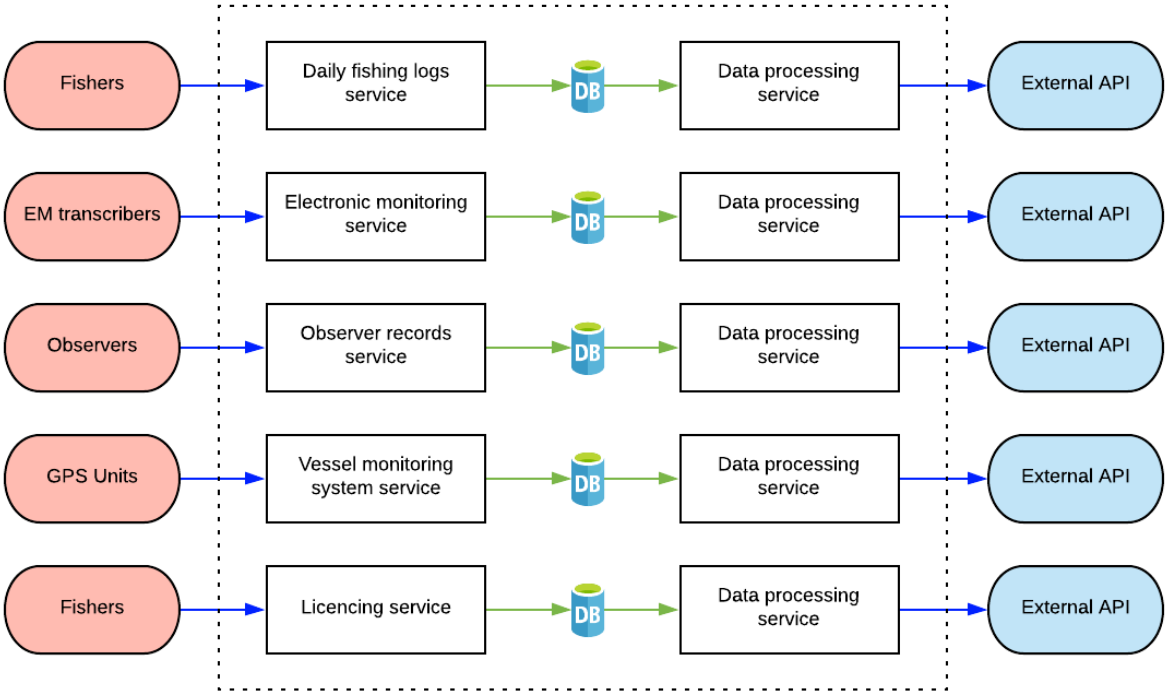


FIGURE 5: DATA COLLECTION AND PROCESSING SERVICES



4.4 Standardised data collection

We found that data is currently received through a variety of methods including paper-based landing report forms, paper-based observer records, electronic reporting of logbooks, video capturing and annotations and VMS GIS data. These methods often differ in accuracy and how quickly the data reaches the agency. This also means that data is delivered in different formats and with differing architectures.

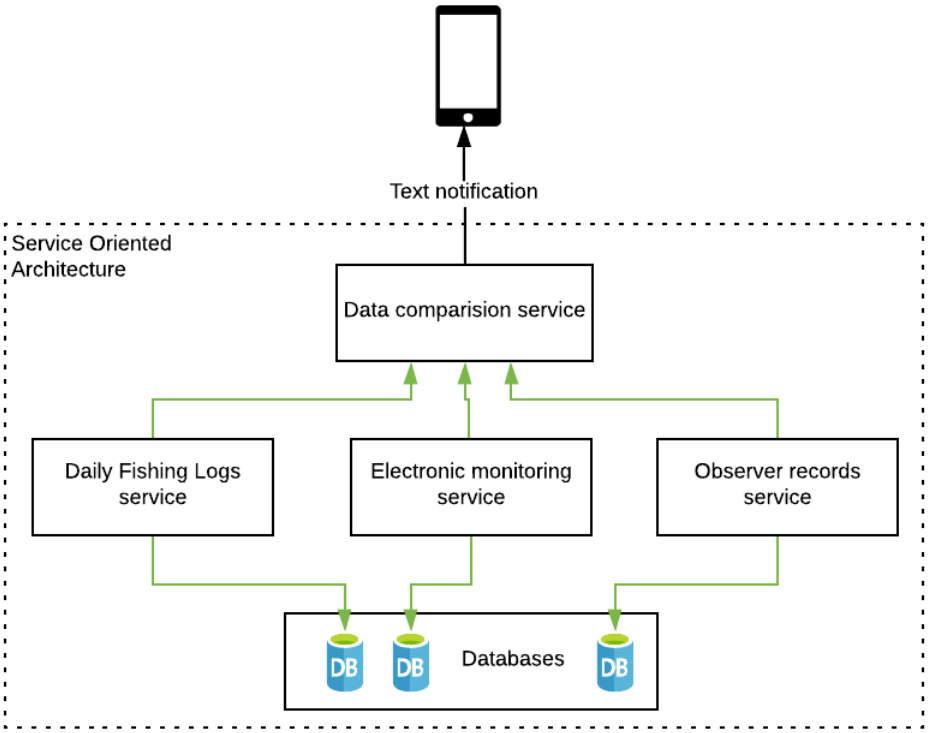
Based on the findings from stakeholder interviews, a number of key challenges emerged:

- Difficulties in collating and comparing data sources due to the different collection methods. Additionally, there is often a significant delay between each data source reaching the agency.
- Data users and fisheries managers often have difficulty comparing data due to irregular collection and data storage formats.
- Data is often received and stored in silos despite being linked. This creates a significant overhead to link, clean, and process the data before any meaning can be drawn.

4.4.1 Using a SOA to unify data collection and increase data accuracy

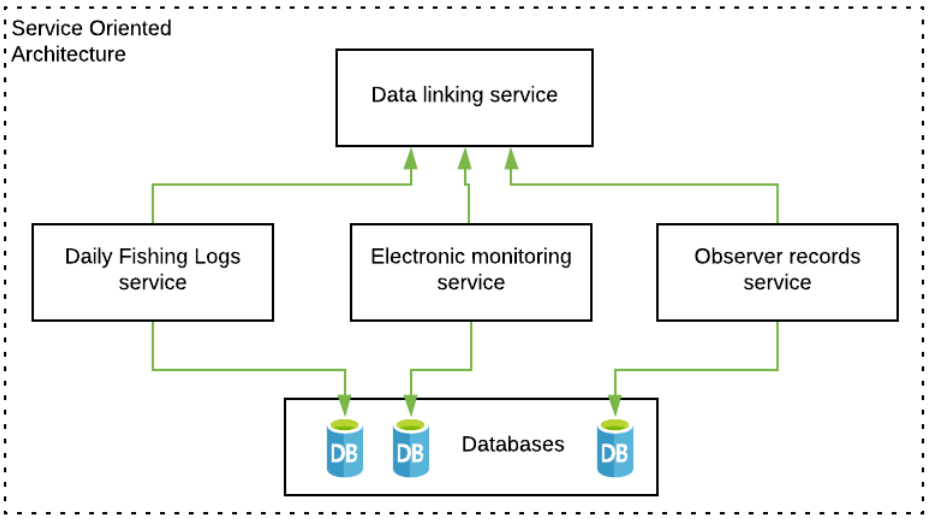
As described in section 4.3, using a SOA will facilitate the creation of services for collecting data. In addition to this, moving to electronic reporting will allow data to be collected in a much timelier manner therefore facilitating comparison between data sources. A SOA will allow incremental delivery of electronic reporting capabilities over time rather than requiring a total overhaul of an enterprise system.

Through linking of data sources, SOA will enable inaccurate, duplicated data sources to be removed. An example of this is the collection of position data in both the fishing logs and by the GPS unit located on the vessel. Rather than relying on fishers to manually report their location when recording fishing activities, the GPS data can be linked with the fishing record based on dates and time, removing redundancies.



The comparison between data sources can be achieved in a SOA by the creation of a standalone service aimed at completing this comparison. This service can be setup to create alerts, emails, or push notifications based on a set of criteria deemed appropriate by the business area responsible for this comparison. Such an example of this could be a text alert to fishers if their seasonal quota has been exhausted, an email notification to compliance if a fisher reports fishing in a protected zone, or text to fishers if discrepancy between reported data such as fishing logs and landing reports is detected.

The setup of such a service is seen in the *SOA with text notification service figure.* If the need arises, data cleaning services can be placed between the data collection services and data linking services.



4.5 System capability fit for purpose

The business needs for fishery management agencies often changes to adapt to new legislation, new fishing methods and new scientific research methods. However, agency infrastructure and software systems are in a constant state of catch up.

Agency infrastructure and software systems have been created in a reactionary way; often in small bespoke parts using outdated technologies.

Based on the findings from stakeholder interviews, a number of key challenges emerged:

- Uplift of the system functionality to meet the evolving business needs is often slow; leading to quicker ad hoc solutions that become the status quo.
- Ad hoc solutions are often poorly integrated further exacerbating the siloed data problem.

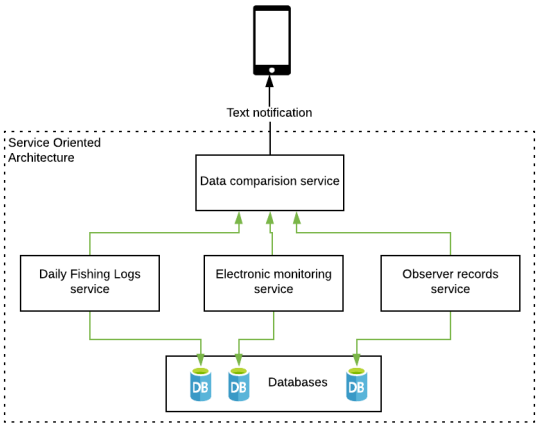


FIGURE 6: DATA COMPARISON SERVICE

- Current system designs do not facility the access and sharing of data in an easy and quick way; this is often done manually.
- Over time many systems have become unsustainable from a support and enhancement standpoint.
- The conglomeration of small bespoke applications causes significant overhead when updating and releasing new features. This often includes large downtime because modular releases can often not be performed.

4.5.1 Using a SOA for the development of appropriate solutions

Utilising a SOA allows services to be created and updated with little effect on the other services. Therefore, when new or changing business processes arise, new services can be easily added to the existing enterprise solution without undertaking an agency-wide uplift of IT systems.

The modular nature of each service allows business users to better understand each system, its components, and the data it deals with. This increased understanding by business users can facilitate clearer requirements when developing a new service or upgrading an existing service.

5. Forms and ADC

During the interview and analysis phase of this project, it was discovered that typically, fisheries management agencies are heavily reliant on forms-based data collection methods. While this is not an essential part of the overall SOA architecture, moving to forms-based storage has several benefits over traditional relational database structures. Forms based storage can best be described as a database that stores each record as its own standalone document. Each document contains semi-structured data that can be queried against using a range of business intelligence tools such as Power BI or through bespoke applications. The benefits of a document store database over a traditional relational database include:

- Ability to quickly handle changing business requirements with regards to what data is collected without requiring an update to a database schema.
- Document Store databases can be scaled horizontally across servers rather than traditional vertical scaling of monolithic relational databases, significantly reducing cost.
- No reliance of SQL, reducing the need for developers to know the inner workings of traditional relational databases.

With these benefits in mind, AFMA has started the development of services using APIs and associated document stores aimed at the collection of logbook data and landing reports as part of the Agency Data Capture (ADC) project.

The e-Fish project, like the ADC project, makes use of Azure services such as function apps and a Cosmos document store database. This approach has allowed AFMA to modularise the reporting methods and also be flexible if they wish to add or remove fields from the forms. The use of the document store database further facilitates this flexibility because it does not rely on a fixed schema.

6. SOA in practice

Figure 7 shows the Service Oriented Architecture being implemented for the proof-of-concept (PoC).

The main objective of the PoC is to introduce the Fishery Management agencies to SOA designs and a range of modern cloud technologies that could be leveraged to meet the Design Principles. Therefore, the proposed architecture aims to "showcase" such technologies and design patterns as opposed to providing a final solution for Fishery Management Agencies to implement.

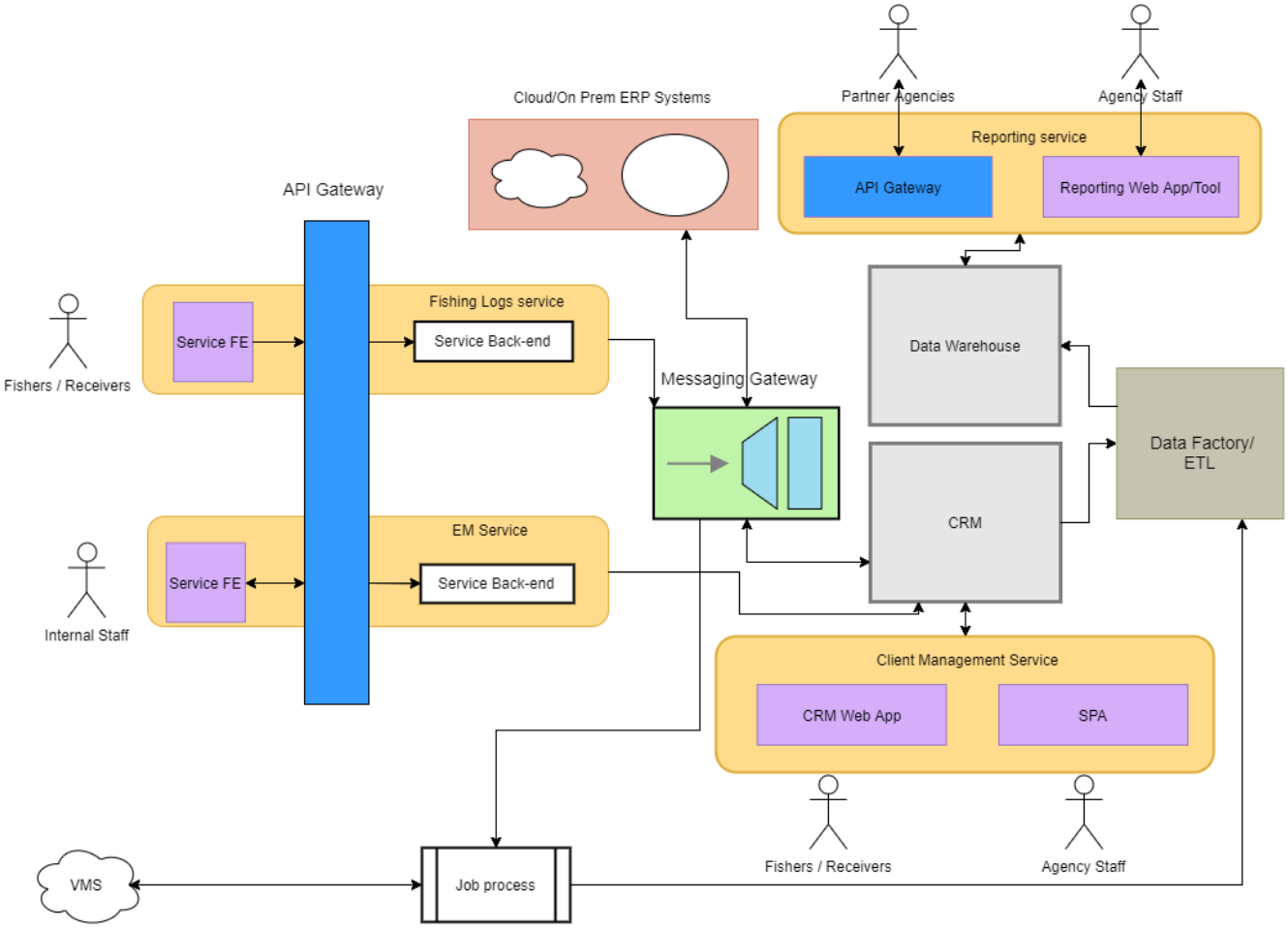


FIGURE 7: POC ARCHITECTURE

Figure 8 illustrates a more in-depth solution architecture recommendation for fisheries agencies, outlining the technology stack and integration between the different services.

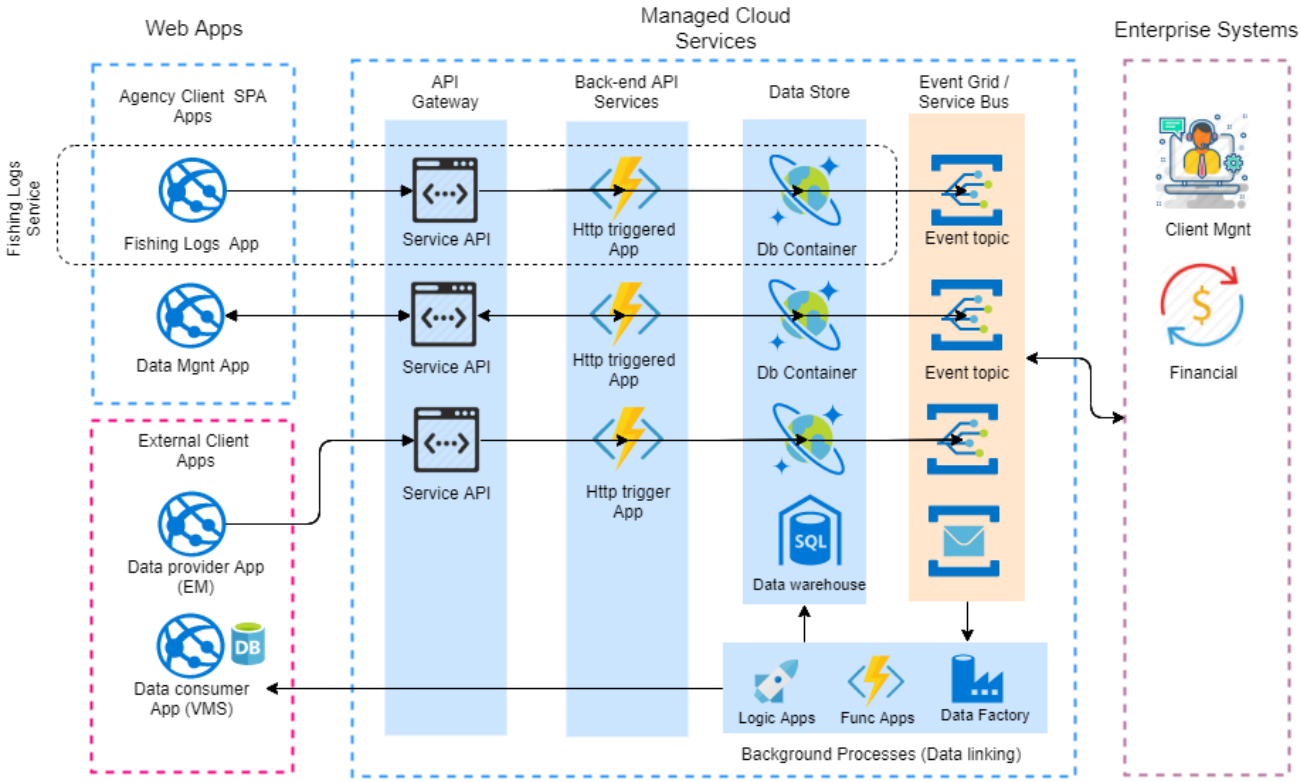


FIGURE 8: SOLUTION ARCHITECTURE DIAGRAM – TECH STACK

The system in Figure 8 has been designed to illustrate how certain data collection and processing services might be designed in a SOA, such as the fishing logs service. This service consists of a web app, managed cloud services including an API gateway, a backend API triggered function, and a cosmos DB. This service is linked to the rest of the agency's ecosystem through the event grid/service bus, and the web app is a UI exposed to external reporting parties such as fishers.

This data is passed through to the backend system where logic is applied to validate the input and transform values (such as code to a human-readable string) before being stored in the database. Upon reaching the database, the event grid/service bus will be triggered; this will facilitate the flow of data through to other systems such as financing or a client management system. Such a setup could also be utilised to facilitate similar business processes such as landing reports or the collection of observer data.

Figure 8 also highlights how systems such as VMS can be integrated with the wider agency ecosystem. This is highlighted in the VMS example. This system has the data provided to the agency ecosystem through a service API, before being processed with a function app and then being stored in cosmos DB. Finally, the event grid/service bus will be triggered to connect this data through the client management application and perhaps a data warehouse. This setup could also be used for any external service provider.

This development model offers the following benefits:

Application Development Strategy

- Both fishing logs and the data management app follow the same pattern with a single page application (SPA) in the front end, an API gateway in the middle and a function app back end (other services omitted for brevity). Establishing a development strategy/pattern is crucial for any development team's general performance and success.
- Enables Rapid Application Development (RAD) by breaking down the components of a given service and distributing the workload amongst the different team members (or teams depending on how big the application is), without having to simultaneously work on the same code, thus minimising source control issues.
- Application can be developed independently, down to the technology or language without having an impact on others or deviating from the overarching development strategy, thus providing more control to the development team.

Document Store

- Document-oriented databases provide great performance when dealing with large volumes of data.
- Since data is stored in a JSON format, it requires no manipulation/transformation from the time it's gathered/produced to the moment it's stored.
- A document container (table) can store documents of different structures; this provides a great deal of flexibility thus making the business better suited to adapt and change.
- In Azure, Cosmos provides native integration with just about every other service, whether it is messaging (e.g. Event grid or Service bus), managed services (APIM or Search) or other services such as Logic Apps and Function Apps.

  This level of out-of-the-box integration is perhaps the single most important advantage of using Cosmos as it significantly reduces the need of developing complex logic/functionality to link data or kick start other business processes.

Single Page Applications (SPAs)

- Developing SPAs using JavaScript frameworks such as Angular or React allows for faster and more rewarding development.
- SPAs can be quite dynamic in nature and purely driven by configuration. In a business domain such a fishery management agency, which is driven by capturing data through forms, being able to update these forms through configuration and capture more data is vital, thus providing business with more flexibility and eases the development burden.

6.1 Fishery management services and scenarios

The PoC uses the following services in demonstrating how data can be integrated in a seamless and efficient manner:

1. Client Management Service (Data Visualisation) – Typically licencing
2. Fishing Logs Service – Typically daily fishing logs
3. EM Service – Typically annotated data of videos
4. Reporting Service

6.1.1 Client Management Service and Data Visualisation

The Client Management Service is depicted as a centralised application where both fishers and agency staff can go about their everyday tasks including:

- Staff can view general information of fishers, vessels, trips and other related data/reports that might come from other services such as EM reports.
- Fishers can renew licenses, transfer quotas, create service requests (case management) or view historic data.

For the purposes of the PoC, the project team decided to implement a Single Page Application to sit on top of Dynamics 365 to showcase business requirements specific to this project.

The Single Page Application was built using the VueJS framework and utilised the open source OpenStreetMap mapping technology. This mapping technology allows users to see points of interest such as "set" locations of individual trips from a boat, heat maps of all "sets" in an area and Commonwealth Fishery Map shapefiles.

Reporting data is displayed on graphs so that vessels reporting catch accuracy can be compared. This data is provided through the API.

6.1.2 EM Service and fishing logs

In the PoC, the EM service mainly focuses on annotated data and how it can be linked to fishing logs, processed (for reporting purposes) and visualised. These services are linked via Event Grid and APIs, which this document goes into more detail in section 6.1.

6.1.3 Reporting service

The reporting service provides a way for agency staff as well as partner agencies to visualise or consume data stored in a data warehouse.

- From an agency point of view, a reporting tool such as PowerBI will integrate seamlessly with any data store and provide powerful features that enable users to create rich and meaningful reports.
- Partner agencies can consume data via an API gateway.

The PoC makes use of a SPA to illustrate some of the reporting options available if data can be reliably linked; in this case, the daily fishing logs and the electronic monitoring data. In the PoC, users are able to view the reporting accuracy of a vessel against the reporting accuracy of the whole fleet over a period of time.

6.2 PoC technologies

The PoC is built using Microsoft technology (Azure and Dynamics 365 CRM) to mock the core services and the integration between them.

NOTE: AFMAs main technology stack is Microsoft, which is why it was chosen for this PoC. However, there are other cloud technologies such as Amazon Web Servies (AWS) or Google cloud platform that could provide the same capabilities.

6.2.1 Azure

Azure is leveraged primarily to demonstrate how data can flow between the different services to ensure consistency and reliability. To achieve this, the PoC uses Azure's main event/messaging services (referred to as the Messaging Gateway in Figure 7). These messaging services are:

- Event Grid
- Service Bus (for ERP systems)
- Logic Apps (workflows)

Other resources such as applications or databases are also be mocked in Azure so that integration between these applications can be demonstrated easily. In reality, these may be on-premises applications or hosted elsewhere.

6.2.1.1 API Management

API Management adds a gateway to other services and authentication which limits what different types of users can see and do. It can also manipulate requests and responses, provides logging and analytics and a developer portal which generates code for interacting with the API and example requests.

6.2.1.2 Azure Functions

Azure Functions are a serverless compute service where code can be triggered from REST APIs, Event Grid events or at time intervals. Azure functions integrate seamlessly with other Azure services and can read/write records to Cosmos DB, publish event grid events or trigger other functions.

6.2.1.3 Event Grid

Event Grid is an eventing backplane that enables event-driven, reactive programming. It uses a publish-subscribe model. Publishers emit events but have no expectation about which events are handled. Subscribers decide which events they want to handle.

Event Grid efficiently and reliably routes events from Azure and non-Azure resources. It distributes the events to registered subscriber endpoints. The event message has enough information for services to react to changes in other services and applications.

6.2.1.4 Service Bus

Service Bus is intended for traditional enterprise applications. These enterprise applications require transactions, ordering, duplicate detection, and instantaneous consistency. Service Bus enables cloud-native applications to provide reliable state transition management for business processes. Service Bus also facilitates highly secure communication across hybrid cloud solutions and can connect existing on-premises systems to cloud solutions.

6.2.1.5 Logic Apps

Azure Logic Apps is a cloud service that helps you schedule, automate, and orchestrate tasks, business processes, and workflows when you need to integrate apps, data, systems, and services across enterprises or organisations.

6.2.1.6 App Service (Web App)

Azure App Services is a container hosting solution for Single-Page Web Applications and APIs and support a variety of popular frameworks like .NET, .NET Core, Node.js, Java, PHP, Ruby and Python.

6.2.1.7 Dynamics 365

The customer relationship management (CRM) application, Dynamics 365, was selected so that the project team could rapidly mock client data and the relationships that live within it. However, it's worth noting that a custom-off-the-shelf (COTS) product like Dynamics 365 plays an important role in SOA as it dramatically simplifies the integration between different systems.

Some of the main features of a CRM product are:

- Extensive out-of-the-box (OOTB) functionality that applies to any industry, providing a cost efficient solution in removing the need for large scale customisation. A great example of this is the self-service portal, where clients can perform basic account management tasks such as updating contact details, raising issues (case management) or renewing licenses in the case of Fishery Management.
- OOTB (or via third-party plugins) integration with other third-party ERP systems such as Financial or HR applications.
- Largely customisable with limited software development time required. The focus on configuration allows for the implementation of custom workflows and plugins, where business logic can be executed.
- Allows emails to be sent and tracked against records.
- Able to integrate with other Azure managed services, such as logic apps and service buses. Allowing the CRM to sync data with other applications (e.g. the eLog Cosmos DB).
- Can perform global search as well as advanced searches on all CRM data. Particularly useful when all the data is sync'd into the CRM, where users can search in one place for information.

7. Design Principles and the proof-of-concept

7.1 Linked Data and Data Integrity

To link data, the PoC makes use of a "Link table" that lives in Dynamics 365. This table will hold a reference to the unique identifiers of other data sets such as client records, fishing logs and EM reports. This approach, while different from using a unique identifier across all data structures (as suggested by the project outline) is similar in nature, but much easier to implement.

This implementation avoids having to copy and maintain the same data in various services, thus reducing the effort towards maintaining data integrity and consistency across all services.

7.1.1 Linking Data in the PoC

This example shows the flow of events that will link a fishing log to a vessel (and other client data), the electronic monitoring (EM) annotated data and the Report Accuracy data. This setup is seen in Figure 9.

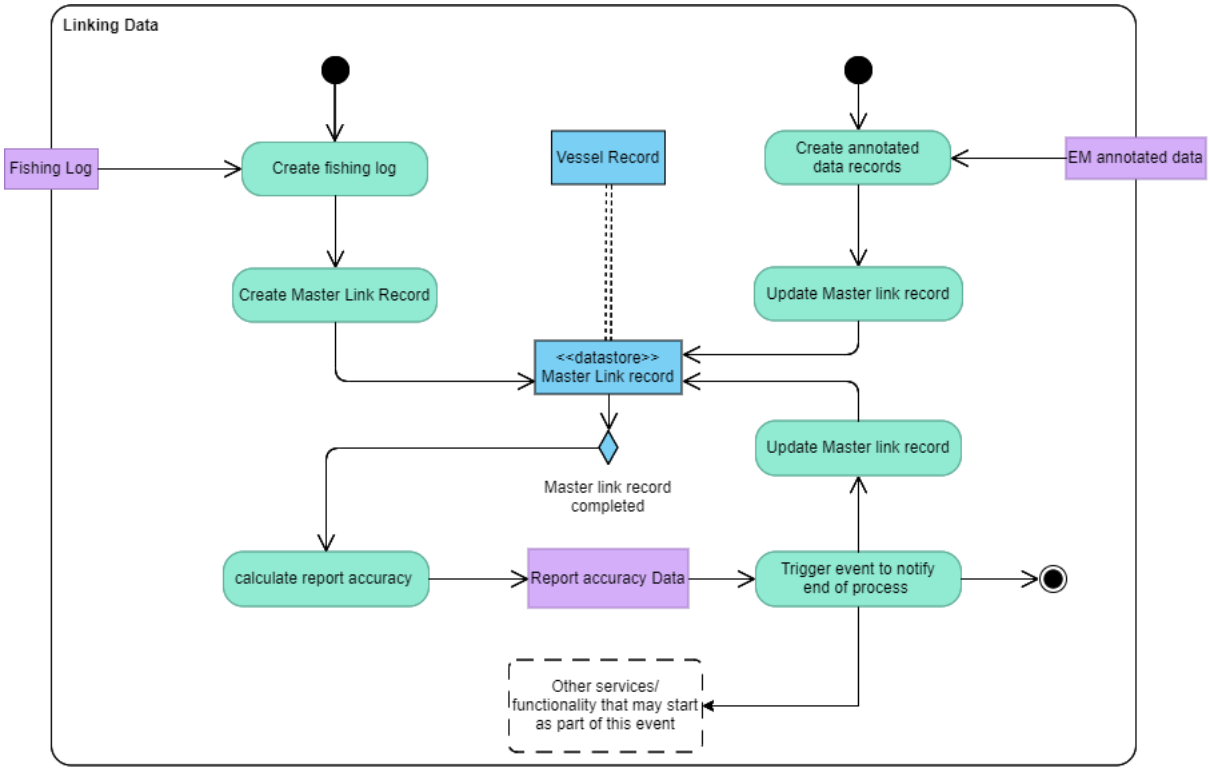


FIGURE 9: DATA LINKING

The process is as follows:

1. A fishing log record is created; this event will create a link record in the master link record table which will automatically link this fishing log record to a vessel, and consequently to a number of entities in the client management service such as contacts (clients), authorities, fisheries etc.

2. The process will be on stand-by until the EM annotated data is added to the system; at that point another event will be fired (using Event Grid), triggering Dynamics 365 to update the link record with the unique identifier.

3. At this point, the system will have the required data to produce a "Reporting Accuracy" record. To produce this record, a dedicated serverless function triggered from Dynamics 365 will collect the necessary information from both E-Logs and EM and produce such record.

4. Lastly, a new event that represents the end of the process can be fired so that other events/processes subscribed to this event can kick-off. The list below shows some fictional examples of how this event could be leveraged:

    - A Logic App workflow can send a completion e-mail notification to whoever is interested in knowing when reports are ready.

    - A Function App process can compare the results of the "Reporting Accuracy" record against historic data and issue a notification to Fishers if the report is below the quality required by the agency.

7.2 Modern Data Sharing & Standardised Data Collection

A Web API Gateway using Azure APIM will be established to allow external partner agencies to consume data. APIM provides a wide range of features that make the integration and consumption of such data very easy, and thus reduces the overhead for both partner agencies as well as fisheries management agencies when it comes to data sharing.

In a similar fashion, the collection and consumption of data produced by agencies, clients or third-party service providers will happen via the APIM Gateway, therefore maintaining a standardised way of collecting and sharing data.

7.2.1 Data Sharing in the PoC

The PoC exhibits this design principle in two ways, the first is sharing fishing data in a FLUX like format and the second is exposing the Dynamics 365 details to external users through an API.

External uses can consume the FLUX formatted data through an API. This API is back ended by a function that converts the eLogs data (stored as JSON) into the FLUX format in XML. This XML is then returned to the user making the API call. The exposed data is customised to the user, in the case of the PoC, no catch details are returned based on a header in the call. This illustrates a number of modern data sharing principles:

1. Data can be collected in any format and shared in another. This means fisheries management agencies can integrate with other agencies using a common standard without having to have to rework their entire data collection and storage process.

2. The exposure of different data based on users means that a single service can be utilised across a number of different user bases. Therefore, duplication issues are unlikely to be encountered.

7.2.2 Standardised Data Collected in the PoC

The PoC makes use of AFMA's agency data capture project to collect the eLogs data. The agency data capture (ADC) project was the uplift of the existing elogs service to a SOA type setup that makes use of APIs and document store databases to capture line fishing daily logs.

The PoC is not able to fully illustrate this concept due to the scope of the project. However, if the PoC was to be expanded to also collect observer data through a dedicated observer service, then this data should also be collected through APIs rather than another format such as paper or spreadsheets.

7.3 System capability fit for purpose

Currently the AFMA on premises licencing and client management applications, Pisces, is an amalgamation of different apps fitted together which requires significant resourcing to maintain a robust system to maintain data.

7.3.1 System capability through use of a CRM

The recommendation to use the CRM application Dynamics 365 to replace Pisces proved to both be fit for purpose and could replace the functionality of Pisces.

Dynamics 365 allows minimal custom code development with OOTB functionality and customisation options available. It also allows for enhancements and bug fixes to be pushed into production more quickly.

Dynamics 365 can centralise all the data and provide search functions (i.e. Global Search and Advanced Find) to the users to find all data from Vessels, Trips, Applications and Invoices. Dynamics 365 on-line also seamlessly integrates with other Azure managed services such as Logic Apps, Function Apps, and Service Buses.

The PoC demonstrates how we can retrieve e-Log data from the Cosmos DB and sync it with Dynamics 365. It also shows how Dynamics 365 can send out messages to a service bus to inform other applications that the Vessel data has changed.

Additionally, the current GoFish website, an AFMA online licencing and fishing industry portal could also be replaced with the use of Dynamics Portals, which comes as an extra service of Dynamics 365 and allows you to create a website which directly connects with Dynamics 365.

8. Lessons learned

8.1 Dynamics 365

Dynamics 365 was originally scoped to simply mimic AFMA's licensing and client management system for the proof of concept. The scope of the system was to create and manage the relationships between the entities (boats, concessions, and authorities). However it was also found during the PoC that Dynamics 365 also provided a simple integration interface that allowed it to be configured with other Azure managed services. This meant that expensive development would be circumvented in favour of simple configuration. This significantly reduces the cost of integrating systems because bespoke development work is not required.

Upon further investigation, Azure managed services such as Logic Apps and the cosmos database could easily be integrated with Dynamics 365, which greatly increased the functionality of the system beyond the initial scope. It was found that Dynamics 365 was able to sync all data from vessels, authorities, concessions, and trip data from cosmos, through configuration. To achieve the same result with a bespoke product would incur a large development cost. Therefore, this out-of-the-box linking functionality allowed the proof of concept to demonstrate how, through the use of Function Apps, Logic Apps and Service Buses, data can be pushed and pulled from the Dynamics 365 system.

It was also found that Dynamics 365 provides a feature to create portals which uses Dynamics 365 as its data store. This opened the opportunity to replace the existing external facing fisheries management portals, such as the online licencing systems, with a custom portal that could quickly integrate with the Dynamics 365 system.

8.2 Data Linking

In order to make production eLogs and EM data available to the PoC, SQL scripts were created to transform relational data from AFMA's electronic monitoring database into JSON data. To make matching simpler the EM JSON was structured in a similar way to ADC eLogs data. Timestamps within the eLogs and EM data were used to match data together and a third linking table was created to record the result of the matching.

For the majority of EM records an accurate match was produced. However, the quality of both EM data and eLogs data was found to be inconsistent; for EM data, records would be incomplete and the eLogs data contained mistakes. This resulted in a number of records that could not be matched (approximately 33% of records were successfully matched) however this was found to be sufficient for the purpose of the PoC. Moving forward from a PoC to a functional system, more investigation would need to take place to have a higher match rate between records.

9. Glossary of terms

Term	Description
Alpha stage	Alpha is the experimental stage of a project. It's an opportunity to use prototypes to work out the right thing to build
API	Application programming interface
COTS	Commercial off the shelf. A product which works "straight out of the box" and allows various customisations
Discovery stage	The initial stage of a project, aimed to get a deep understanding of the problems users are trying to solve
EM	Electronic monitoring
FLUX	The Fisheries Language for Universal Exchange
OOTB	Out of the box. A product or feature that works without any or minimal customisation
PowerBI	A business analytics service by Microsoft
REST	Representational state transfer – A type of web service
RFID	Radio-frequency identification, commonly used on credit/bank cards
SBR AU	Standard Business Reporting
VMS	Vessel monitoring system