DRAFT
E-Enterprise and Exchange Network API Product Management Framework

April 2020
Version 2.0

Abstract

APIs are powerful connectors that can deliver access to data or functionality. They can help agencies efficiently develop and operate applications, move data, and provide capabilities within and between organizations. With the trend toward the use of APIs expected to accelerate, environmental agencies need an overarching API vision and a shared management framework to fully take advantage of this technology and support powerful collaboration. The purpose of this document is to provide an initial version of an E-Enterprise API Management Framework.

Revision History and Acknowledgements

Revision History

Version	Date	Comments
1	2/13/20	Baseline Version
2	4/6/20	Updated Baseline Version

Acknowledgements

This paper is a product of the E-Enterprise Digital Strategy Team, which is comprised of the following members:

- Vince Allen, U.S. Environmental Protection Agency
- Mary Curtis, U.S. Environmental Protection Agency
- Matt Leopard, U.S. Environmental Protection Agency
- Kari Jacobson Hedin, Fond du Lac Reservation – Region 5
- Andy Putnam, Colorado Department of Public Health and the Environment

Table of Contents

List of Tables

List of Figures

1 Overview

The *E-Enterprise and Exchange Network API Management Framework* provides E-Enterprise partners with an overarching Application Programming Interfaces (APIs) vision and a shared management framework to fully take advantage of APIs and support powerful collaboration on their joint development. E-Enterprise partners will develop APIs in different ways and for different uses. The framework does not seek to standardize API development but instead identifies common areas for sharing best practices that support the development of APIs that can be used across partners.
The current version of this document reflects some unknowns that will be filled in over time as E-Enterprise partners provide feedback, share best practices and participate in collaboration opportunities. In addition, the document contains recommendations to continue the buildout of the framework and its implementation.

The primary audiences for the framework are technical development managers and agency staff who support the software, systems, and data flows that enable interoperability and interactions across environmental programs and agencies.

1.1 The API Framework and the Exchange Network

The Exchange Network was built using Web Services, which at the time was the best format to exchange large batches of data in a secured flow. As business processes and technology have evolved, the Exchange Network has been working to expand the tools available to the community including APIs and associated standards so they can be applied by E-Enterprise partners. This API Framework is a first step in that direction. Work on this API framework will be complemented with a parallel joint effort to reassess the evolving business needs and develop a plan for existing Exchange Network tools and the business functions they fulfill. This will allow partners to match the best technologies and designs to the Exchange Network's evolving business needs.

Exchange Network and API Framework Recommendations for Moving Forward:

1. Validate the Exchange Network business functions and the design patterns they support to develop plans for modernizing Exchange Network data flows, standards, and tools.
2. Reflect knowledge gained from the API Framework into the Exchange Network Grant Program.

1.2 API Challenges, Opportunities and Goals within the E-Enterprise Community

Environmental agencies in the E-Enterprise community are increasingly turning to APIs to enable many of their most critical business interactions. APIs are powerful connectors that can deliver access to data and functionality. They can help agencies efficiently develop and operate applications, move data, and provide capabilities within and between organizations. With the trend toward the use of APIs expected to accelerate, environmental agencies need an overarching API vision and a shared management framework to fully take advantage of this technology and encourage collaboration.

This document was developed in response to the following identified challenges:

- E-Enterprise partners are developing new APIs to exchange data (e.g., updating facility identification) without any mechanisms for cross-program and cross-agency coordination and learning. It is the speed and flexibility of API development which makes them so powerful; however, without coordination, this speed and flexibility could lead to the need to build the information and silos E-Enterprise aims to reduce.

  Opportunity: Leverage the experience of industry best practice and the E-Enterprise community's experience to develop an API framework for collaboration and innovation.

- There are many operating models[1] over which APIs are developed and used by E-Enterprise partners. There is no one-size-fits-all solution to API development.

  Opportunity: E-Enterprise API guidance will apply differently and with different levels of rigor, depending on business requirements. For example, security and message standards for secured regulatory exchanges between partners will have different requirements from public APIs used solely to access data.

- Building services with APIs requires more than just the adoption of new standards on topics such as syntax or documentation.

  Opportunity: Develop high-level agreements to address the challenges of existing Exchange Network flows and many broader API application opportunities. These are discussed in *Section 2: Service Design and Lifecycle Management*.

- API management tools and platforms are rapidly co-evolving with the associated standards. The capabilities of these tools may shape which standards E-Enterprise will adopt and how they are implemented.

  Opportunity: Standards must be tool neutral as different partners will choose different tools for API management. Pilot work led by the New Mexico (NM) Environment Department is an example of the use of an API management tools (Refer to Appendix B: E-Enterprise Partner Use Cases for more information on the NM Use Case).

Based on the challenges, the **goals** of the E-Enterprise API Framework are to:

- Support the E-Enterprise Digital Strategy (Digital Strategy) goal of developing a successful ecosystem of APIs between E-Enterprise Partners and their users, including APIs interacting with shared core regulatory systems.
- Affirm APIs as a complementary and core technology of the Exchange Network and its ongoing evolution.
- Create and support an E-Enterprise community of practice for all types environmental APIs in which there is shared interest to provide cross-program API knowledge sharing and act as an integration point for API work that spans functional areas and organizations. This community of

[1] Operating models includes who has the authority for a given environmental program and who is hosting the core software used to support it.

practice will be supported by a collaboration site in which members can exchange ideas, seek support from each other, find answers on processes and tooling and find guidance and standards.

- Leverage opportunities to improve interoperability among applications, systems, domains, and organizations as new systems are being developed and legacy systems are being re-engineered.
- Identify opportunities for how API standards, technologies, and platforms can be leveraged by the E-Enterprise community.

1.3 Overview of the API Management Framework

The *E-Enterprise and Exchange Network API Management Framework* serves as the foundation for designing, building, managing, and sharing APIs – in short, the entire lifecycle of the API process. Developers often refer to API frameworks as code blocks for programming language, which allow the integration of services from a third party to send emails and fetch data using the third party's API. However, this document expands this definition to include suggested methods, principles, and components.

The API Management Framework is a set of principles and methods intended to guide E-Enterprise Partners in planning, building and publishing APIs using best practices. The framework promotes alignment, collaboration, and delivery employing the underlying principles used by product delivery teams. It borrows from the pillars of customer-centric product development and leverages three primary bodies of knowledge: lean agile software development, product development, and continuous delivery. In addition, the design of the framework incorporates lessons from the history of the Exchange Network, existing environmental API efforts, and other API strategy literature.

This document is organized by the framework as depicted in *Figure 1: API Management Framework*. The framework is anchored by the following principles: Service Design and Lifecycle Management (Section 2), API Management Tools (Section 3), and Standards and Shared Tools (Section 4). The two outer pillars support the principles and the sharing of information through the collection and sharing of E-Enterprise Partner use cases (Appendix B) and the Collaboration site, which will host information for an API E-Enterprise Partner Community of Practice[2]. The overall framework consists of the exchange of knowledge, best practices, technology trends, standards, shared tools and use cases.

[2] The Community of Practice (CoP) and Collaboration site are in concept form only. The establishment of the CoP and collaboration site have yet to be determined.

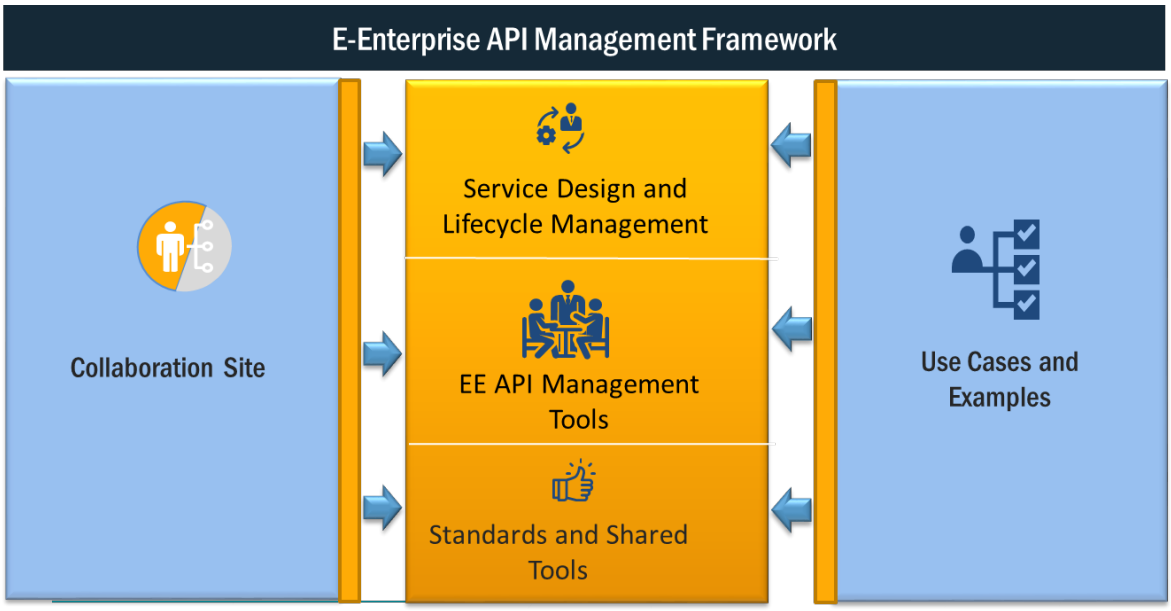

Figure 1: Enterprise API Framework

As the framework is further developed, standards and guidelines will be developed in the following key areas. These components are described in more detail in Section 4: Standards and Shared Tools:

- **Service Model:** Implement a customer-centric approach to identifying business needs and the data exchange flows.
- **General API Design:** Ensure the design and construction of individual APIs align with the APIs that make up the service.
- **Change Management:** Ensure that the entire lifecycle of APIs is managed with a deliberate approach, including version management, communicating changes to users through product roadmaps and deprecating APIs when it is time to retire them with appropriate processes.
- **Developer Onboarding:** Provide guidance to help developers quickly learn, test, and use APIs.
- **Documentation:** Provide standards and documentation around how APIs can be shared and consumed.
- **Authentication and Authorization:** Use industry standards and best practices for authentication and authorization.
- **Monitoring and Observation:** Provide transparency into how APIs operate after being published.
- **Discovery:** Ensure that published APIs are discoverable.

The following sections provide more detail about the principles outlined in the API Management Framework—Service Design and Lifecycle Management; E-Enterprise API Management Tools; and Standards and Shared Tools. Each section contains a set of recommendations to continue the buildout and implementation of the framework.

2 Service Design and Lifecycle Management

As described in Section 1.1, the API management strategy literature and experience indicate that developing "API Standards" is important, but alone it is not enough. This section introduces two broader strategic concepts that are a key part of this framework:

- **Service Design** is the term for the principles and methods collectively needed to plan out high-level functions of the core business services before any API is developed.
- **API Lifecycle Management** is the adaptation of the industry-standard model for how APIs are managed from early conceptualization through retirement and deprecation. It also includes the key concepts of "API-as-Product" and "Product Ownership." Both of these concepts and the methods used to support them have great application for E-Enterprise partners' shared APIs.

API Data Streams

The scope of the API will differ based on groups involved in the data exchange. The following are the potential data streams, which are not mutually exclusion.

- Partner to Partner Exchange
- Regulated entities reporting to one or more E-Enterprise Partners
- Open Government/Public Access
- Internal Agency Applications

Appendix A: API Data Flows contains more information about each API data stream.

These concepts are discussed below. While these concepts are still very preliminary, it is intended to introduce the broad concepts and introduce the structure of the next round of development of this framework, which will include guidance based on feedback collected from E-Enterprise partners.

2.1 Service Design

Service Design is the strategy and design of services that will be operating between E-Enterprise Partners to support a business need. This design happens at the business or programmatic level and should be vetted before any individual API is developed. Service Design is important because data exchanges, flows or application integrations that do not satisfy a business need within the context of an operating model generally cannot be fixed with APIs, regardless of how well designed the individual APIs are. Lower level API standards are important, but alone are insufficient to get to the next generation of integration and joint capabilities envisioned in the Digital Strategy.

For APIs operating between agencies, some of the most critical decisions impacting interoperability will be made in the high-level design of the national systems' services operated by EPA. This connection of "API First" has already been made indirectly by the E-Enterprise Leadership Council (EELC) whitepaper "Operationalizing the Principles of E-Enterprise and the Digital Strategy," which calls for a new approach to program modernization.

Illinois/EPA NeT Application Facility Data Integration Use Case

Illinois is planning to receive and manage Notices of Intent through the EPA hosted NeT Web application and maintain control of the facility data created through the process. To accomplish this, EPA and Illinois are exploring an integration pattern that will use a semi-automated process, including a REST API between the state and EPA. Appendix B: E-Enterprise Partner Use Cases contains more detail about this use case that illustrates application integration across partners and the co-management of key data.

This API Framework includes a complementary component (described below) called "Service Modeling" that provides common integration and exchange patterns and models to leverage partners' collective experience, as well as other best practices to make service design easier.

Once a service design has been iterated and adopted, Lifecycle Management examines how the collection of APIs need to be managed through a standard set of lifecycle phases from early conceptualization through retirement and deprecation. The following sections look more closely at the concept of API-as-a-Product and the role of API Product Owner.

2.1.1 Service Models

There are common integration and exchange patterns among partners that can be cataloged and developed into options for how they can be supported with services. In addition, there are documented lessons learned as exchanges and APIs have been developed. Many of these lessons are broadly relevant. There is also significant literature around service design in government[3] by groups like 18F as well as the private sector.

This component highlights the importance of these shared models, patterns, and design principles. They are at a higher level than those in the individual API design component but are more specific than the general approaches of the Service Design layer.

2.2 API as a Product

API-as-a-product is a term for how to approach design thinking, prototyping, researching, and testing APIs. E-Enterprise Partners should treat their APIs as products rather than pieces of functionality because the APIs are services that are being consumed by downstream machines or people that need the data. Consequently, there are decisions that need to be made about the services that partners are going to provide. The basic questions that make up good design include:

- What is the problem to be solved?
- Who are your customers?
- Do we know what your customers (machine, public) need?
- How will your customers use this data?
- What type of documentation or onboarding is needed for them to be successful?
- What is the post-life management needed for this API?
- Once it has been published, what are the anticipated care and maintenance needs, and what is the plan to maintain it?
- How many APIs are needed to provide all the functionality needed?

Multiple APIs may be combined into one greater API. This is referred to as an "API Product" or "API Bundling." Depending upon the complexity of the product, partners may consider an API management tool to help modify and create proxies to the multiple APIs. In software development, the Product Owner manages and owns the life of the APIs.

[3] Work by the Facility team included an application of a set of standard "plays" developed by 18F and their application to the development and design of the Facility services.

2.3 The Role of Product Owner

An API Product Owner is not the programmer, but instead has the regulatory background to understand current and future data user needs and domain workflows. Large systems may have a Product Manager with multiple Product Owners and smaller systems may have only one Product Owner.

As part of service design, the API Product Owner connects the programs' goals with the needs of the community and works within the guardrails of their budget to ensure that enterprise architecture, program management, and operations are all in alignment for the proposed solutions. The API Product Owner should have the authority and autonomy to make the key decisions to plan the API work as they are the subject matter expert (SME) who is most familiar with the state, regional and tribal needs as well as regulatory requirements and deadlines. Being close to the community of users and E-Enterprise Partners, the Product Owner understands what is needed now and in the long-term. The Product Owner is best prepared to answer these questions: "Where are we heading? Are we building the right products and services?"

The following are the API Product Owner key roles and areas of expertise. The role of the API Product Owner will be different depending on the scope and operating model of the APIs.

- **Authority to Make Decisions with Autonomy:**
    - Ability to quickly respond to the changing needs of the stakeholders by using continuous feedback loops and informing the API product teams about how the APIs will be used. They prioritize requests and maintain the flow of work that is essential to creating quality product delivery.
    - Prioritize the wants and needs of customers and determine what should be developed now versus in the future.
    - Make decisions about the future of the APIs, including whether to continue to enhance and improve capabilities or discontinue a service if it no longer adds value.
- **Point of Contact**: The Product Manager for big systems and the Product Owner for smaller systems is the main point of contact for the system that is being designed/ built for which there will be APIs.
- **Customer-Centric**: The API Product Owner is responsible for understanding the needs of the community and ensuring that the development team is building the right services and delivering the correct data to serve the customers. This requires the ability to:
    - Understand and prioritize the customers' needs.
    - Communicate effectively with the fast-moving API community and serve as both a tester and researcher.
- **Technical Expertise:** The Product Owner has a technical mindset but is not necessarily a developer. They are passionate about the product. They understand security requirements on published data and the need for centralized API product security.
- **National, State, Regional or Tribal Regulatory Expertise:** The API Product Owner has a role that is similar to what was previously known as the point of contact for the data exchange node owner, but with more knowledge about how the exchange of information between E-Enterprise Partners will improve if it can be done in smaller packages and closer to real-time. Changes that are made to the flow in which data is reported or exchanged have consequences, both from a business process perspective as well as technological implications.

In summary, the Product Owner role has a broad set of responsibilities and is critical to the success of a program that is doing transformative work toward API-as-a-Product that will improve business services to E-Enterprise Partners. Without this single point of authority that can pivot and make quick decisions, maintain the flow of work, and act as the voice of the customer, there is a high risk of developing the wrong APIs and experiencing project failure.

Service Design and Lifecycle Management Recommendations for Moving Forward:

1. Recommend the EPA CIO establish API Product Owner roles for external APIs of EPA National Systems.
2. Conduct a half-day Product Owner Workshop in Fall 2020 to provide an overview of product owner roles, roadmap development best practices, and collaboration best practices.
3. Build out the Service Design and Lifecycle Management Pillar through the collection of best practices from EPA National Systems.

3 E-Enterprise API Management Tools

The API Framework consists of a set of API Management Tools, which provide the community with the capabilities to plan, design, implement, collaborate, and communicate about the work they are doing on API development.

E-Enterprise API Collaboration Site

The API Framework will be supported by the development of an E-Enterprise API Collaboration Site[4], which will host the following:

- E-Enterprise API standards, guidelines, and best practices documentation
- A discovery tool for environmental APIs
- Use cases and examples of best practices in operational environmental APIs
- An interactive API Community of Practice (CoP), particularly for cross-program issues and for communities without collaboration space
- Product roadmap for shared APIs
- A testbed and possible shared hosting for supporting API Management functions, including gateway, security, and related functions. Many of these are related to the functions above.

New Mexico Shared API Management Platform Pilot

New Mexico is working with the E-Enterprise Federated Identity Bridge to integrate a shared API management platform using a commercial tool (Google Apigee). State and EPA staff are experimenting with the capabilities, including managing permissions, managing proxy and orchestration; managing API security integration; hosting dynamic documentation; and providing capabilities for a full featured developer portal. Knowledge from this pilot will be leveraged for an E-Enterprise API Management and Collaboration space. Refer to Appendix B: E-Enterprise Partner Use Cases for more information.

[4] The E-Enterprise API Collaboration site is in concept form only. The architecture and management of a collaboration site have yet to be determined.

API Product Roadmaps

API Product Roadmaps inform the community of APIs that are being planned for current and future development. Roadmaps are an important tool for many APIs and are especially important for the EPA national systems. Roadmaps not only provide transparency but also an opportunity for alignment with future planned development. They create consistency with the new business processes and functions and the evolution of transition plans as organizations move from their current business to the new API business processes.

With the key inputs—solution vision, budgets, and prioritized lists of requests—the Product Owner will work with their team to create a plan and an **API Product Roadmap**, which provides information about what is planned to be released as well as what is forecast to be developed in the near future. These roadmaps will be shared on the E-Enterprise API Collaboration Site. The site's community of users will use these roadmaps to obtain a better understanding of what is currently under development as well as what the Product Owners forecast to meet the needs of their communities. Roadmaps allow API users to plan their work around the potential availability of upcoming services and data streams. *Figure 2: API Product Roadmap* provides an example of an API roadmap and depicts what the Product Owner is committing and forecasting in the next 6 months. The forecast does not commit the Product Owner to any API deliverable dates but indicates the APIs that are intended to be developed under favorable conditions.

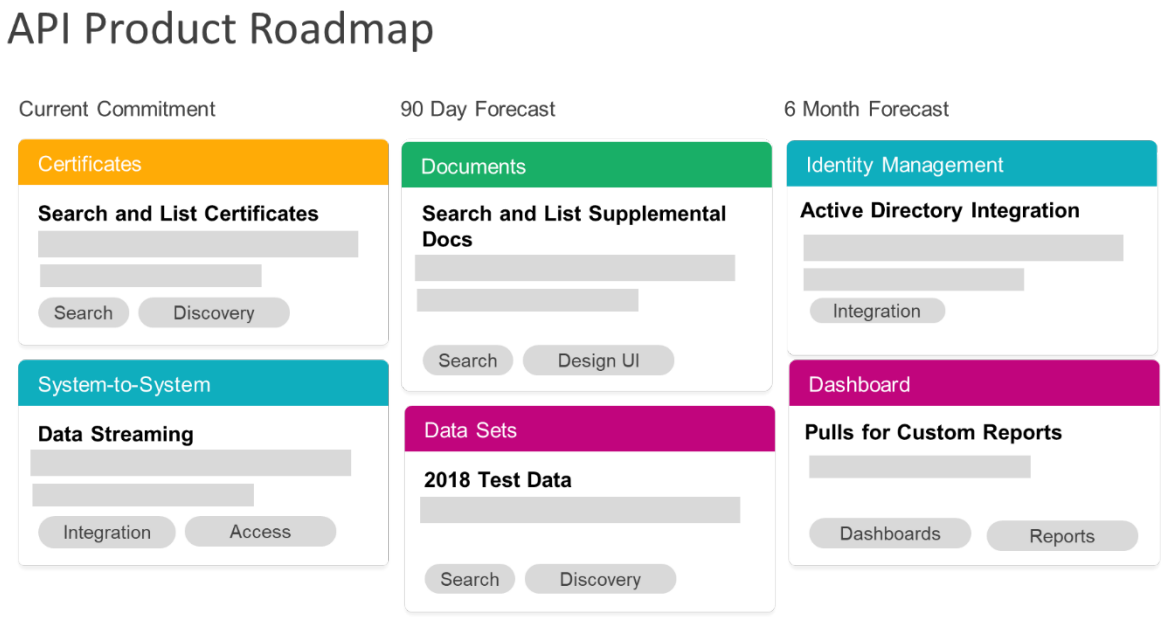

Figure 2: API Product Roadmap

Community of Practice

A Community of Practice will be formed for E-Enterprise API Product Owners to share lessons learned and outcomes from the development of their APIs. This will help with the ongoing development of best practices and standards for the E-Enterprise partners. As new standards and services evolve and other E-

Enterprise Partners test services, additional best practices, use cases, lessons learned, and documents will be made available to the community of practice.

Use Cases

As E-Enterprise Partners develop their API-as-a-Product, they will post their roadmaps, models, designs, and solutions to the collaboration site. E-Enterprise partners are encouraged to implement this E-Enterprise API Framework; demonstrate good service design principles; and inform the Community of Practice of plans, use cases, and best practices. The sharing of stories, success and most importantly, the lessons learned are important to the Community of Practice. These should be written as use cases and posted to the Collaboration site. (see Appendix B: E-Enterprise Partner Use Cases for example use cases).

E-Enterprise API Management Tools Recommendations for Moving Forward:

1. As part of the Product Owner Workshop, provide best practices and templates for creating API Roadmaps.
2. Conduct an analysis of existing API Collaboration Spaces, such as New Mexico and the Department of Veteran's Affairs. Based on the analysis, develop a business case for an E-Enterprise API Collaboration Space.
3. Create a Community of Practice to support Product Owners.

4 Standards and Shared Tools

Standards play a supporting role in this proposed framework; however, case studies from industry indicate that large organizations that overreached with standards or created an overly centralized approach to API management did not perform better on quality or delivery.[5] The E-Enterprise API Framework focus is to grow relevant standards by building on the Exchange Network community to develop a broader environmental API community of practice that will share knowledge and best practices. The best way to communicate successes and lessons learned is to collect and provide examples from early adopters and innovators in the E-Enterprise community.

API Standards and Guidance

E-Enterprise API standards and guidance will be available on the E-Enterprise API Collaboration Space. The standards will evolve and be influenced by the community of practice's best practices. The standards and guidance could lead to a new, powerful technology stack layer where policies are based on how standards are managed, implemented, and enforced. For example, on the E-Enterprise collaboration site, E-Enterprise partners with differing native local APIs could share and use a unified API approach by using a common gateway proxy, mapping, and brokering capabilities. Different partners

[5] *Continuous API Management – Making the Right Decisions in an Evolving Landscape*; M. Medjaoui, E. Wilde, R. Mitra, M. Amundsen, O'Reilly Media, Inc 2019

will choose different tools for API management, so standards must be tool-neutral. Pilot work in New Mexico is demonstrating these connections and opportunities. (See the NM use case example in the Appendix B: E-Enterprise Partner Use Cases.)

Common Standards for Components and Capabilities

Table 1: API Management Framework Components and Capabilities provides a high-level summary of the components and capabilities of the pillars of the E-Enterprise API Management Framework and identifies opportunities for guidance and standards. These categories have been gleaned from many literature and government sources[6], including previous work done by EPA and the Department of Veterans Affairs.

Common standards for components and capabilities add value by:

- Providing preferred options from which to draw upon as new services are design and built between EPA, states, and tribes.
- Support a rich Community of Practice that requires a collaborative investment from all partners in aggregating, curating and promoting the platform.

EPA E-Manifest API Implementation

EPA has implemented E-Manifest with a strong emphasis on enabling integration through an extensive bundle of APIs. E-Manifest is one of the first major EPA programs to implement secure reporting at this scale and they were faced with implementation decisions in all the supporting API component and standards areas, especially security, documentation and developer onboarding. Refer to Appendix B: E-Enterprise Partner Use Cases for more information.

E-Enterprise API Standards Recommendations for Moving Forward:

1. Form a small team to work through the standard components in Table 1 to:
    a. Identify existing standards/guidance and identify the key types of operational issues involved in their adoption/implementation.
    b. Recommend a prioritized list of additional standards/guidance for development.

[6] *API Design and Management Guidelines.* Prepared for the EPA Office of Information Management, Information Exchange Solutions Branch by CGI. September 26, 2019

Table 1: API Management Framework Components and Capabilities

Components	Need	Opportunities for Guidance and Standards
Service Model	How can we design services that work better for individual customers and interoperate when needed? How do we ensure the services fit with a reasonable Concept of Operations for our major shared systems?	• Codify exiting work on common data exchange patterns and service designs that have worked well. • Develop a set of "Design Principles" for EPA National Platforms that harvest lessons learned from current best practices and inform the Service Design component. • Identify high level enterprise data, such as Facility, that requires interoperability at the service level to achieve common goals.
General API Design	How can we design and construct individual APIs that make up a service? This includes syntax of API, versioning, message format, payload validation and other design elements.	• Adopt existing industry standards to improve environmental APIs. • Identify and prioritize the most important elements of APIs for standardization. Identify where standardization is not needed and/or point to reference sites.
Authentication and Authorization	How do we use industry standards to build APIs with appropriate level of course through fine grain permissions?	• Identify authentication and authorization operating models based on the security policies of individual agencies. • Identify where is it important that partners agree to specific security standards and where multiple approaches can co-exist. • Demonstrate where E-Enterprise partners can leverage the experience of leading implementers.
Monitoring and Observability	How can we monitor and have transparency into the operation of an API?	• Identify where monitoring and observability capabilities are important to E-Enterprise vs an individual agency. • Identify opportunities for the Collaboration Site to share and integrate monitoring capabilities.
Discovery	How can we ensure that APIs are easily discoverable?	• Document through use cases how E-Enterprise partners leverage standards, tooling and platforms to create an E-Enterprise space for discovery in the proposed E-Enterprise API Collaboration Site. • Identify how E-Enterprise partners can future proof the distributed discovery model that the future environment will create.
Documentation	How do we provide standards and examples of good documentation and shared infrastructure to support documentation created for discovery and consumption?	• Identify how the E-Enterprise can adapt OpenAPI standards and leverage tooling for dynamic documentation and other capabilities (such as client generation).
Developer Onboarding	How can we enable developers to quickly learn, test and use APIs?	• Identify how the E-Enterprise API Collaboration Site can be used to "self-service" for rapid sign-up and self-provisioning.
Change Management	How can we fix, update and improve the catalog of APIs as quickly as possible without breaking anything? How do we communicate these changes to our community of users? How do we implement changes that may be requested from other regulated partners that impact our API product roadmap?	• Develop a change management strategy for communicating the implementation and versioning of APIs; advertise API release schedules; and communicate problems and fixes to the community.

Appendix A: API Data Flows

APIs are used in many ways by many E-Enterprise partners. This section provides the preliminary approach to scoping and categorizing APIs in order to further develop a common framework to manage them. The framework will apply differently and may provide different value depending on these data flows. These data flows focus on areas of mutual interest to partners and their interoperability.

Data Flow 1: Direct Regulatory Partner Use and Interoperability

APIs are used by multiple agency partners, via external interfaces, in direct support of major regulatory and non-regulatory programs. This includes:

- Traditional EN flows between and among states, EPA, and tribes
- Existing and new program or domain-specific APIs in use or in development between partners (see the NeT use case in Appendix B)

Data Flow 2: EPA National Systems Workflows and Business Logic

This data flow could be considered a part of data flow 1, but is separated due to the design and operation of EPA national systems having a large impact on partners through many channels—direct users, data providers and consumers, and co-regulators in complex workflows.

As API architectures become the standard, the service architecture decisions regarding internal interactions will impact downstream external-facing services providers. Many of the issues with interoperability stem from basic service architecture and system business rules in these systems. As illustrated by the NeT use case example (Refer to Appendix B: E-Enterprise Partner Use Cases), there are many scenarios where partners will want to integrate across their own and other hosted applications. It is anticipated that there are many more diverse types of interaction patterns between partners, well beyond just submission or downloading data.

In the past, the EN provided the tracing of schema changes for modifications to data elements and business rules, but this process has not been updated and may no longer be sufficient given the more complex integrations anticipated going forward.

Data Flow 3: Interactions with Regulated Entities

Regulated entities will report to multiple E-Enterprise Partners. Identifications or business identifiers create interoperability where state-regulated entities also report to EPA (and vice versa) and can benefit from improved coordination among EPA and state reporting streams (such as in CAER, TRI, e-Manifest). Interoperability means several things in this context. It is an improved user experience (UX) for the reporter and an improved implementation option for state and EPA system developers in building applications and managing data with cross-partner regulated entity dependencies.

Data Flow 4: Public Access/ Open Government

Creation and management of public facing APIs is a major part of many agencies. Structure of these APIs is typically determined by the owner agency's standards or major domain standards, such as in the water community. In addition, many agencies have adopted high-level open standards such as Open Data Protocol (OData) and are utilizing open source or commercial Software-as-a-Service (SaaS) tools

such as Socrata for their data publishing portals. This includes EPA's EnergyStar program which hosts a large collection of public APIs using a SaaS platform. These tools support key functions such as security, discovery, documentation hosting, and API structure guidance that are included as components in the proposed framework. The analysis initially focused on Data Flow 1 and 2, but it is clear that the extensive work already done in this area by E-Enterprise partners should be reviewed more closely to share.

Data Flow 5: Agency Internal APIs

The vast majority of individual APIs that will be developed by an agency are internal and tightly coupled to that organizational domain and its applications. These APIs will be governed first by agency standards and guidelines where they exist, so they are not the main focus of this framework. The infrastructure used by EPA to manage internal facing APIs will be the same, or closely related to, the infrastructure for managing external facing APIs. E-Enterprise guidance, literature, recommendations and stories may be useful as peer-learning opportunities from fellow agencies. As in other areas, and programs within agencies, will be a few steps ahead of others in capabilities like API management. Their examples may be useful as a starting point. This framework is influenced by the work of these federal and state agencies.

There are many other data flows yet to be developed including peer agency collaborations and interactions with other types of major data providers and consumers such as academia.

Appendix B: E-Enterprise Partner Use Cases

E-Enterprise partners are encouraged to implement this E-Enterprise API Framework; demonstrate good service design principles; and inform the community of practice of plans, use cases and best practices.

This section includes a few use cases that demonstrate aspects of the practices and methods described in this framework.

New Mexico Shared API Management Platform Pilot

New Mexico is the lead state on an EN grant to continue their work with the E-Enterprise Federated Identity Bridge to integrate a shared API management platform using a commercial tool (Google Apigee). State and EPA staff are experimenting with the capabilities of this extensive platform, including its capabilities to:

- Manage permissions so that API owners can manage their own APIs securely and easily provide permissioned access to them;
- Proxy and perform compositing and orchestration;
- Manage API security integration with the bridge;
- Host dynamic documentation; and
- Provide capabilities for a full featured developer portal including interacting with hosted APIs.

States plan to do a proof of concept API integration using the shared gateway effort to build maps which draw dynamically on state APIs and the Tableau Server tool hosted by the State of Colorado to build maps displaying border state environmental information.

EPA staff and contractors were also provided an account and were able to test the platforms features to support API development and hosting.

Why is this use case important?

The New Mexico pilot inspired and informed two related themes of this framework:

- The opportunities, needs and challenges of a shared API Collaboration and Management Platform to enable both community learning and the potential for federated management of APIs and the kind of capabilities they could enable.
- The close coupling of API Management Platforms (or a federated network of them) with the underlying standards for capabilities like security, documentation, and developer portals over the entire lifecycle of APIs.

Knowledge from this and other pilot work could be leveraged through an E-Enterprise API Management and Collaborative Space to provide a first-generation version of such a capacity to interested E-Enterprise Partners.

State EPA NeT Application Facility Data Integration via API

The State of Illinois has delegation for the National Pollutant Discharge Elimination System (NPDES) program. It also has a local system for managing its integrated facility data, which includes the NPDES program. Illinois plans to use the EPA hosted NeT program to receive and manage Notices of Intent (NOIs) through the EPA hosted NeT web application. In addition, Illinois seeks to maintain control of the facility data created through the NOI process. To do so EPA and Illinois are exploring an integration

pattern that will use a semi-automated process, including a REST API between Illinois and EPA, which will help meet this business need.

Why is this use case important to the API Framework?

First, the use case illustrates the program context in which service design should happen:

- A state with authority for a regulatory permit program is using part of an EPA hosted web application but seeks to integrate that process and associated data with its locally hosted system. API calls between the agencies will be used for this integration. It is expected that this pattern will grow increasingly common.
- Facility data which are, in effect, "co-managed" by the State of Illinois and EPA are a form of "enterprise data" that require dedicated business processes and procedures to manage effectively. In this case both a manual and automated process is needed to meet the business need. APIs fulfill part of this need.

These two factors—application integration across partners and authorities, and co-management of key data—are common to many of the business processes partners manage.

Second, this use case illustrates the kind of business needs for which we expect API's to be applied. The Illinois technology leadership's preliminary discovery process to work through the constraints and possibilities (e.g., security of inbound and outbound APIs) are similar technology and management issues that E-Enterprise partners will encounter. Standard approaches will be required if EPA and partners are going to scale these types of interactions across more and more processes.

EPA E-Manifest API Implementation

As authorized by Federal Statute, EPA has implemented E-Manifest with a strong emphasis on enabling integration through an extensive bundle of APIs. This is one of the larger traditional media transactional programs, which includes extensive security, digital signature requirements, and complex relationships with state manifest authorities.

Why is this use case important?

E-Manifest was one of the first major EPA programs to implement a secure reporting API at this scale and complexity. Consequently, they were faced with implementation decisions in all of the supporting API components and standards, especially security, documentation and developer onboarding, including:

- Use of Active GitHub for developers
- Transparent public web presence during early development work (Trello) which migrated an active GitHub
- State access/integration of manifest data

Appendix C: Developer-Focused Guidance and Information

The information contained in this section will be housed in specific developer's guidance documentation moving forward. Until those documents are started the information is housed here for continuity and to help with referencing.

Service Model

Service Model describes how well our services meet the applicable criteria and standards in order for them to be made available to E-Enterprise Partners. Key principles of the Service Model include.

- Designate one person with the authority and responsibility to assign tasks and make business, product, and technical decisions. The API Product Owner is ultimately responsible for how well the service meets the needs of its users, ensuring that service features are built, and the work is managed as planned. This concept is addressed in Section 2.1.1 of this Framework.
- Begin with the end product in mind and iterate through the entire process. Understand the different ways people will interact with the API services. This can be done by understanding why your E-Enterprise Partners want the data, and how the API will help make tasks easier or solve problems.
- Understand the needs of your customers using a model-based approach. Design an initial interface model and test the design from the E-Enterprise Partners perspective. Exploring and pinpointing service users' needs will inform technical and design decisions. This is the basis for a customer-centric approach.
- Build services that are simple and intuitive enough that users succeed the first time, unaided in adopting them. Simplicity–the art of maximizing the amount of work not done l – is an essential Agile principle that applies API development. Keep it simple and uncomplicated.
- Consider designs that can be deconstructed into individual functions and methods, which are easier to use and change, rather than one large complex API.
- Use RESTful APIs and document and develop prototypes that can be deployed in a sandbox to test the API functionality.
- Use event-based interfaces to publish data from your services so it can be used for reporting and analytics. Ensure that all services manage data relevant for analytics and reporting through use of a consistent, event-based interface style and event broker.
- Design the API to be consistent in nomenclature, methodology and documentation related to other APIs. Ensure that common data fields are consistent across service interfaces, including event-based interfaces. Data-related interactions between services will break down without shared, consistent definitions of key data fields. Using the standards and guidance provided by the E-Enterprise API Framework can help adhere to this Service Model principle.

General API Design

How can we design and construct the individual APIs that make up a service? This includes syntax of API, versioning, message format and other design elements. This section covers basic design constructs. How can we adopt and extend existing industry standards to improve environmental APIs? What are the most important elements of API to standardized, and where does it not matter?

- REST syntax
- message, data and payload formats,
- endpoint and service naming
- version indicator
- error code usage

Authentication and Authorization

This section describes the preferred solutions and guidance for API security. It provides preferred solutions and guidance for API security for general use cases. There is consensus on mechanisms such as tokens and API keys.

Monitoring and Observation

Providing observability of Scope 1 API to consumers and providers. Include "health checks" of key API's.

Discovery

How do we ensure that critical API are discoverable? This overlaps with developer onboarding, and documentation. Well-described catalog of APIs.

Documentation

All APIs must be documented using the approved interface definition language (IDL) and the Open API specification. API documentation must be validated to ensure it adheres to the specification and must reflect methods (POST, GET, DELETE, PUT) and operations for the services that API exposes. The details of OpenAPI are included in the Standards.

Development teams implementing APIs should consider the following when documenting APIs:

- Consider adopting design practices by creating the API documentation during initial development. After iterating the API definition, the service is implemented and documentation is refined. Creating the documentation in advance increases the probability of building a service that meets the needs of clients.
- Consider using API documentation markup tools (e.g., Swagger editor, Swagger Hub), mocking services, code validation tools, and code generation tools.
- Consult the API standards established by the General Services Administration's (GSA) 18F Group[7] and the guidance developed by the EPA Office for Information Management. [8]
- Consider the purpose of the API (e.g., presentation, orchestration, system) and its uses in the initial design and documentation of the API. See General API Design in Section 4.2 of this document for more guidance.
- Consider the privacy implications of the API and develop a privacy statement describing the purpose, uses, and types of PMI data that will be processed by the API. The design and use of

[7] Refer to API standards established by the General Services Administration's (GSA) 18F Group at https://github.com/18F/api-standards.

[8] *API Design and Management Guidelines.* Prepared for the EPA Office of Information Management, Information Exchange Solutions Branch. September 26, 2019

data **must** be consistent with the original purpose of the information collection; any secondary uses **must** be described.

- Consider design practices (e.g., 18F API Standards) when developing API documentation. Consider accommodations for including relevant metadata. The documentation should include sufficient metadata to enable clients to calculate total data and determine when and how to fetch the next set of results. See Figure 3 for an example from 18F.

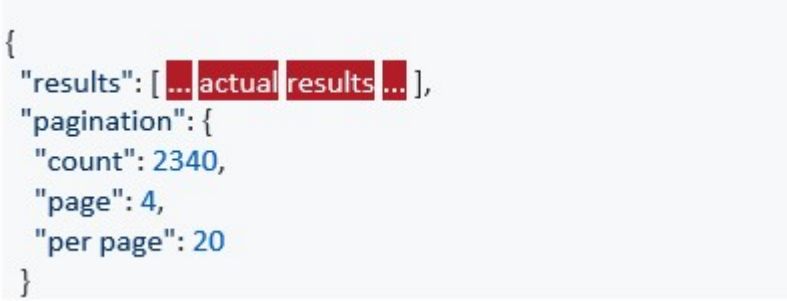

Figure 3: 18F API Standard Showing Metadata Documentation for Pagination

Key Attributes

Table 2 displays a set of attributes that are applicable for OpenAPI specifications that are based on the Representational State Transfer (REST) architecture style.

Table 2: Minimum Attributes[9]

Attribute	Description	Open API Field
API Name	A name for the Application Programming Interface (API)	Info Object
Description	A clear explanation of the function of the method/resource; this should be a description of the API's purpose, functionality, value, and user community	Info Object
Version	Current API version	Info Object
URL	API endpoint	Info Object
URL Parameters	A list of parameters used on this resource/method, including types, special formatting, rules, and requirements	Path Item Object
Endpoints and Operations	Describes methods (e.g., GET, PUT, POST, DELETE, OPTIONS, HEAD, PATCH, TRACE) and operations to invoke methods	Path Item Object
Security	Authentication (e.g., HTTP basic authentication, API key, Open Authorization (OAuth), OpenID Connect)	Security Requirement Object

[9] *API Documentation*, VA Facilities: https://devapi.va.gov/services/va_facilities/docs/v0/api

Attribute	Description	Open API Field
Access Type	Internal/External, Trusted/Anonymous	External Documentation Object
Response Codes	HTTP Status and Error Codes	Responses Object
Samples/Examples	• A sample call, with correlating media type and body • A sample response, including media type and body • Code examples for multiple languages, including all necessary code • Software development kit (SDK) examples (if SDKs are provided) showing how to access the resource/method; and how to use the SDK for the language	External Documentation Object
Points of Contact (PoC)	• Business PoC: The primary sponsor representative for the API that provides requirements and guidance on decisions that impact business functions; and that are supported by the API • Technical PoC: Responsible for day-today management of the API; ensures that all technical system components (software, infrastructure, platforms, network, security) are operational and integrated to support successful API functionality	Info Object for Technical PoC; External Documentation Object for Business PoC

Document APIs in OpenAPI Format

Developer teams should follow the Open API specification format to document APIs. The development teams should follow the most recent OpenAPI version.[10] An overview of the Open API document schema is provided in Table 3.

Table 3: Open API Document Sections

OpenAPI Field	Data Type	Example	Note
openapi	String	openapi: 4.6.2	Required by OpenAPI Specification
info	Info Object	See Section 4.6.2.1	Required by OpenAPI Specification

[10] The most recent OpenAPI version is available at https://github.com/OAI/OpenAPI-Specification.

OpenAPI Field	Data Type	Example	Note
servers	Servers Object	See Section 4.6.2.2	
paths	Paths Object	See Section 4.6.2.3	Required by OpenAPI Specification
components	Components Object	See Section 4.6.2.4	
security	Security Requirement Object	See Section 4.6.2.5	
tags	Tags Object	See Section 4.6.2.6	
externalDocs	externalDocs Object	See Section 4.6.2.7	

The following subsections provide details on each of the main sections of the Open API format. A more detailed reference of the Open API specification is found in the Reference Section (Appendix A) of this document.

Set Info Object

The Info Object must contain metadata to define the description, point of contact (PoC), and other information. The following example illustrates fields in this object for an Open API Specification definition of an API.

Example [11]

```
{
   "info": {
      "title": "Example App",
      "description": "Veteran experience application to show aggregated view of
Veteran information.",
      "termsOfService": "https://developer.va.gov/explore/terms-of-service",
      "contact": {
         "name": "John Smith",
         "url": "https://developer.va.gov/explore/exampleveteran",
         "email": "veteransupport@va.gov"
      },
     "license": {
         "name": "CC BY-SA 4.0",
         "url": "https://creativecommons.org/licenses/by-sa/4.0/"
      },
      "version": "1.0"
   }
}
```

[11] VA Developer Portal at https://developer.va.gov.

Set Server Object

The Server Object defines connection data for the basepath of the server that hosts the API and service. The following provides a simple example. Additional server variables, such as username and port, can also be specified.[12]

```
{
  "servers": {
    "description": "Health record system API example",
    "url": "http://examplesystem.epa.gov/exampleAPI/"
  }
}
```

Set Paths Object

Follow Open API specification to set Path Item Object for all relevant endpoints and operations.

This object contains the main details of RESTful methods (e.g., GET, PUT, POST, HEAD) for the API. An abbreviated example is provided in this section. When documenting all methods and operations, project teams are expected to have a much larger Path Item Object.

```
{
  "paths": {
    "/results": {
      "get": {
        "description": "Returns all results the system can access",
        "responses": {
          "200": {
            "description": "A list of results",
            "content": {
              "application/json": {                    "schema": {
                  "type": "array",
                  "items": {
                    "$ref": "#/components/schemas/resultsdisplay"
                  }
                }
              }
            }
          }
        }
      }
    }
  }
}
```

[12] For additional details on fields that can be specified, refer to the Open API Specification at https://github.com/OAI/OpenAPI-Specification.

In this example, $ref refers to the Component Object, defined by the JSON reference specification. This is used to access features that may be used across several API definitions or resources. For security reasons, some platforms may implement authentication and authorization to restrict access to the Paths Object.[13]

Set Components Object

The Components Object defines reusable content in the specification. This object can provide schemas, responses, parameters, examples, requestBodies, headers, securitySchemes, links, and callbacks. The following is an abbreviated example of a component object.

```
{
  "components": {
    "schemas": {
      "error": {
        "message": "The system encountered an error."
      }

    }
  }
}
```

The following is an example of a reference to a component definition of this error schema.

```
{

 "responses": {

  "NotFound": {

    "description": "Resource not found."

  },

  "GeneralError": {      "description": "General error",

    "content": {

      "application/json": {

       "schema": { "$ref": "#/components/schemas/error"

        }

      }

    }

  }
```

[13] Refer to the Open API Specification, version 3.0.2, Security Filtering Section, at https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.2.md#securityFiltering.

```
 }
}
```

Set Security Scheme Object

Refer to Section 'Authentication and Authorization' for information on this object

Set Tag Object

The Tag Object adds metadata to a single tag used by the Operation Object. It is not mandatory, but it can be used to group content in API tools, as in the following example.

```
{
  "tags": {
    "name": "Customer PII",
    "description": "Customer Personally Identifiable Information"
  }}
```

Set External Documentation Object

The External Documentation Object references external resources that provide extended documentation (e.g., SLA, example code). The following provides an example of using the External Documentation Object to point to additional information.

```
{
  "externalDocs": {
    "description": "API Examples and Sample Code",
    "url": "https://developer.va.gov/exampleveterancode"
  }
}
```

Separately, the Operation Object within the Paths Object can include links to external documentation for a given operation. The Tags Object can also include links to external documentation.

Developer Onboarding

How we enable developers to quickly learn about, test and use API's. Includes concept of "Self-service" for rapid sign up and self-provisioning. The E-Enterprise API Collaboration Site provide might provide capabilities such as a developer sandbox, common standards and guidance.

Change Management

Will provide recommendations on how to manage decisions, being transparent, version management and releasing at a reasonable rate. Example of an API Roadmap. Using the Collaboration Site to publish the roadmaps.

Appendix D: References

18F GSA API Standards: https://github.com/18F/api-standards

Example API Documentation, VA Benefits Intake: https://devapi.vets.gov/services/vba_documents/docs/v0/api

API Documentation, VA Facilities: https://devapi.va.gov/services/va_facilities/docs/v0/api

OpenAPI specification: https://github.com/OAI/OpenAPI-Specification

REST Cookbook: http://restcookbook.com/

OAuth 2.0 Framework: https://tools.ietf.org/html/rfc6749#section-1.2OAuth 2.0 Threat Model: https://tools.ietf.org/html/rfc6819#page-16

Continuous API Management – Making the Right Decisions in an Evolving Landscape; M. Medjaoui, E. Wilde, R. Mitra, M. Amundsen, O'Reilly Media, Inc 2019

VA Enterprise Design Patterns, Enterprise Architecture – Enterprise Service Oriented Architecture (SOA), Office of Information and Technology, Version 2.0, April 2018

Application Programming Interface (API) Enterprise Design Pattern, API Release Standard, Enterprise Program Management Office, U.S. Office Department of Veterans Affairs, February 2019

Overcoming Bureaucratic Barriers to Digital Transformation, CIO Whitepaper - Office of Information and Technology, U.S. Office Department of Veterans Affairs, December 2018

API Design and Management Guidelines. Prepared for the EPA Office of Information Management, Information Exchange Solutions Branch. September 26, 2019