Digital Prescribing & Dispensing Pathways Programme

Physical Architecture Description

DOCUMENT CONTROL SHEET

Title:	
Date Published/Issued:	
Date Effective From:	
Version/Issue Number:	
Document Type:	
Document Status:	
Author:	
Approver:	
Approved by and Date:	
Contact:	
File Location:	

Approvals*:* *This document requires the following signed approvals:*

Name	**Title**	**Date**	**Version**

Distribution:

This document has been distributed to:

Name:	**Title/Board:**	**Date of Issue:**	**Version:**

Revision History:

Version	**Date**	**Sharepoint Version**	**Summary of Changes**	**Name**

Contents

DPDP Physical Architecture Introduction

Introduction

Purpose

The purpose of this document is to describe the physical architecture of the digital prescribing and dispensing pathways (DPDP) service, referencing an overview of the business architecture that the solution must support and realise in order to successfully deliver the DPDP programme objectives.

The physical architecture is presented as a series of perspectives, each containing a number of views. Each perspective broadly groups together related views that describe particular areas of architectural concern.

Audience

The primary audience of this document are those stakeholders and other parties who have interests related to the overall technical solution. These stakeholders include internal NSS and NES teams, potential and incumbent suppliers, technical team members from organisations with system dependencies on DPDP or ePharmacy, and Scottish Govt technical strategists.

The secondary audience of this document are those stakeholders with an interest in the overall delivery of DPDP or services with a cross-dependency on the programme. This includes programme leads, business stakeholders, healthcare leads, sponsors, external agencies, suppliers and Scottish Govt healthcare strategists.

Scope

The scope of this document covers the technical solution, including how it realises the business capabilities. It does not include the business capabilities or processes with the context of the DPDP programme; these are described in a separate document. Other services outwith the DPDP domain may be referenced within the context of the solution if there is a dependency between the process/service/system in question and digital prescribing.

In scope for the DPDP solution architecture are:

- Advanced Electronic Signature (AES) signing of prescriptions.
- Public Key Infrastructure (PKI).
- API-centric integration of services and components within DPDP, including orchestration and choreography of underlying services, data transformation, service resilience.
- Notifications service for alerting patients of prescription events.
- API access control and management.
- New integration routes (“pipelines”) into existing systems as part of intermediate transition architectures.
- User registration and authentication for AES.
- Public access via a portal (possibly as a mobile app) for prescription-related information.
- Direct messaging between prescribers and dispensers.

Out of scope

- Network or domain authentication
- Changes to 3rd party systems, such as PMR’s, GP IT, beyond identifying the API operations needed to support those services
- Changes to downstream systems other than to provide pipelines to feed data into those systems.

- Identity and access management (IAM) beyond the core requirements to authenticate prescribers for AES.
- Analysis of the incumbent ePharmacy (aka ePMS) messaging-based prescription service from Atos. This analysis is available in the document DPDP - Analysis of ePMS.docx [1].
- Design of the business architecture required for the successful delivery of the DPDP programme. The business architecture is available in the document DPDP Business Architecture.docx [2].

Structure of Architecture Documents

This architecture documents are broken down as follows:

Section	Purpose
Introduction	This section contains the purpose, audience and scope of the document.
Business Architecture	Contained within a separate document [2]. That document contains the capability, actor and process views of the business architecture, including a breakdown into activities and value streams.
Solution Architecture	Contained within a separate document [9]. That document contains the logical views of the solution architecture, including overall vision and contextual views, the more detailed description of each subdomain and its services, and dynamic views showing collaborative behaviour between services where appropriate, within the solution domain.
Physical Solution Architecture	This section contains the physical views of the solution architecture, including the reference architecture that all new services within the DPDP will be expected to adhere to. The implementation technologies and platforms are described within the physical domain.
Solution Architecture Mapping to Business Architecture	Contained within a separate document [9]. That document maps the logical solution architecture to the capabilities, value streams, functions and processes within the business architecture [2].
Data Architecture	Contained within a separate document [10]. That document contains the logical and physical views of the data architecture, including data flows, state models, data schema and data-tier integration platforms.
Integration Architecture	Contained within a separate document [9]. That document contains views of the integration architecture where additional information is required beyond the descriptions in the preceding sections.
Deployment and Infrastructure Architecture	Contained within a separate document [11]. That document contains the deployment views, including availability and service continuity aspects, as well as the environment chain to be used within the DevOps delivery pipeline.
Delivery Phases	Contained within a separate document [9]. That document contains views of the logical architecture for the known delivery releases from MVP to target end-state architecture.

Section	Purpose
Appendices	This section contains supporting information and further detailed breakdowns.

References

[1] Analysis of the incumbent ePMS service: DPDP - Analysis of ePMS.docx (Sharepoint permissions required), current version

[2] DPDP Business architecture design: DPDP Business Architecture.docx, (Sharepoint permissions required), current version

[3] DPDP Architecture Decision Log: DPDP Architecture Decision Log.xlsx, (Sharepoint permissions required), current version

[5] DPDP NFR Requirements Log (draft): DPDP NFR requirements.xlsx, (Sharepoint permissions required), current version

[7] DPDP Architecture Principles: DPDP Architecture Principles v1.0.docx, (Sharepoint permissions required), version 1.0

[8] DPDP Business Glossary: Business Glossary.xlsx, (Sharepoint permissions required), current version

[9] DPDP Solution Architecture Description: DPDP Solution Architecture Description.docx (Sharepoint permissions required), current version

[10] DPDP Data Architecture Description: DPDP Data Architecture Description.docx (Sharepoint permissions required), current version

[11] DPDP Deployment and Infrastructure Architecture Description: DPDP Deployment and Infrastructure Architecture Description.docx (Sharepoint permissions required), current version

Physical Architecture

Physical Architecture of Solution

The physical architecture of the solution for the DPDP service is broken down into the physical architecture of each of the domains defined in the solution architecture. There is no overall physical architectural perspective due to the complexity of the whole DPDP domain in terms of platforms in use and their deployment environments. Refer to the Overall Logical Solution Architecture section of [9] for logical views of the entire DPDP service

Solution Architecture Domains

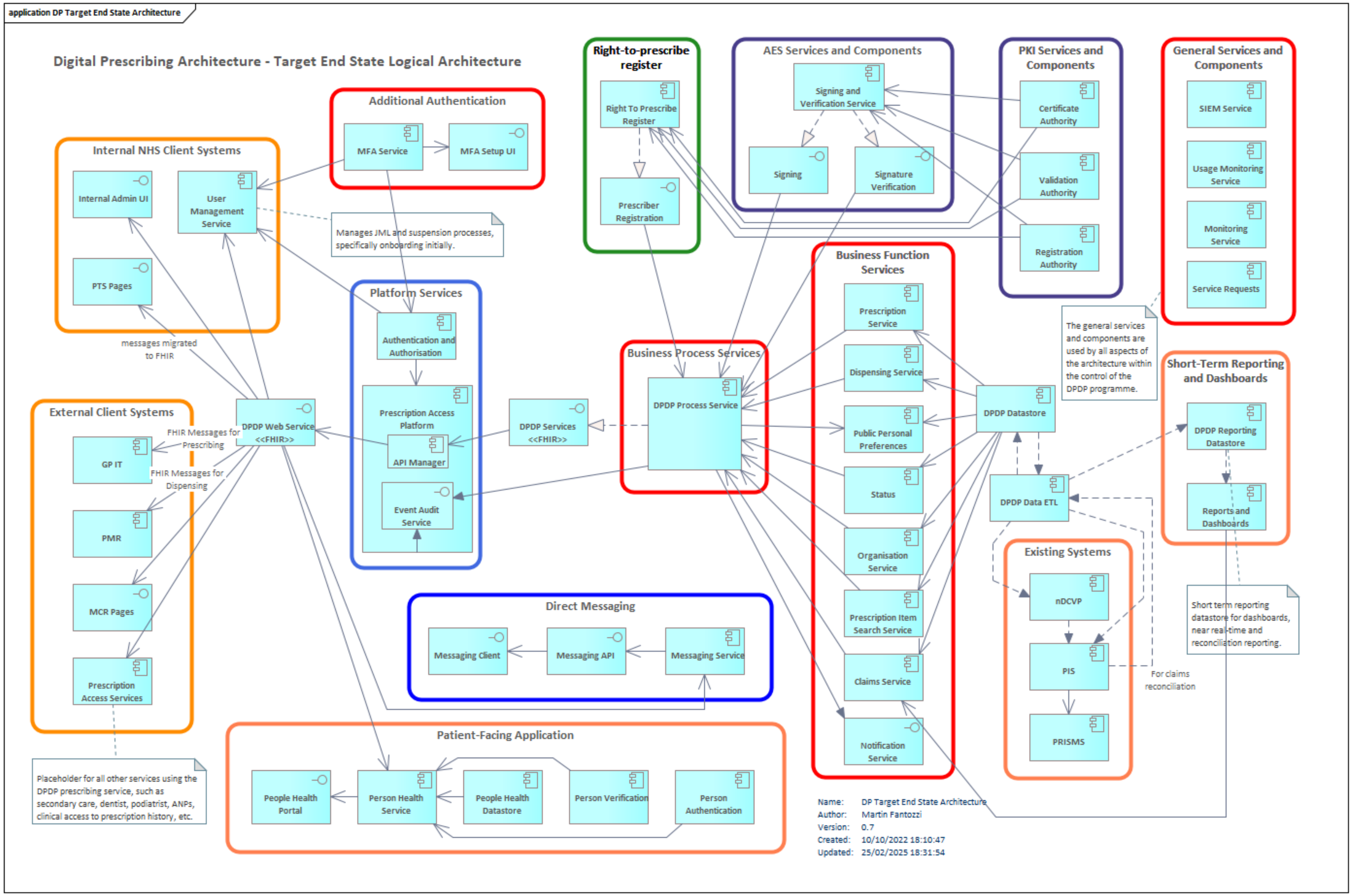
The following sections contain a number of physical views of the solution architecture within each of the key DPDP domains.   In some cases the domains are collated into a larger section. The architecture is based on containerised micro-services, and there is a reference architecture section that provides views of the patterns and platforms that are in use for those micro-services, with all domains that are dependent on that reference architecture containing a link to the relevant section.

Note that there are a number of critical architectural decisions within the overall DPDP solution architecture. Where they have been made the physical architecture will reflect any impact of that decision and references made to the decision log [3] that contains a record of that decision and the relevant options paper that supports it if one exists.

As the architecture of each domain evolves, the physical architecture and logical deployment architecture will be updated. The overall deployment architecture is provided within [11]. Business terms are defined in a central business glossary [8].

Overall Solution Architecture

The following diagram provides the overall logical solution architecture for DPDP, refer to [9] for further details



Manages JML and suspension processes, specifically onboarding initially.

The general services and components are used by all aspects of the architecture within the control of the DPDP programme.

Short term reporting datastore for dashboards, near real-time and reconciliation reporting.

Placeholder for all other services using the DPDP prescribing service, such as secondary care, dentist, podiatrist, ANPs, clinical access to prescription history, etc.

Name: DP Target End State Architecture
Author: Martin Fantozzi
Version: 0.7
Created: 10/10/2022 18:10:47
Updated: 25/02/2025 18:31:54

Reference Architecture for Platform Services, Business Processes, Business Functions

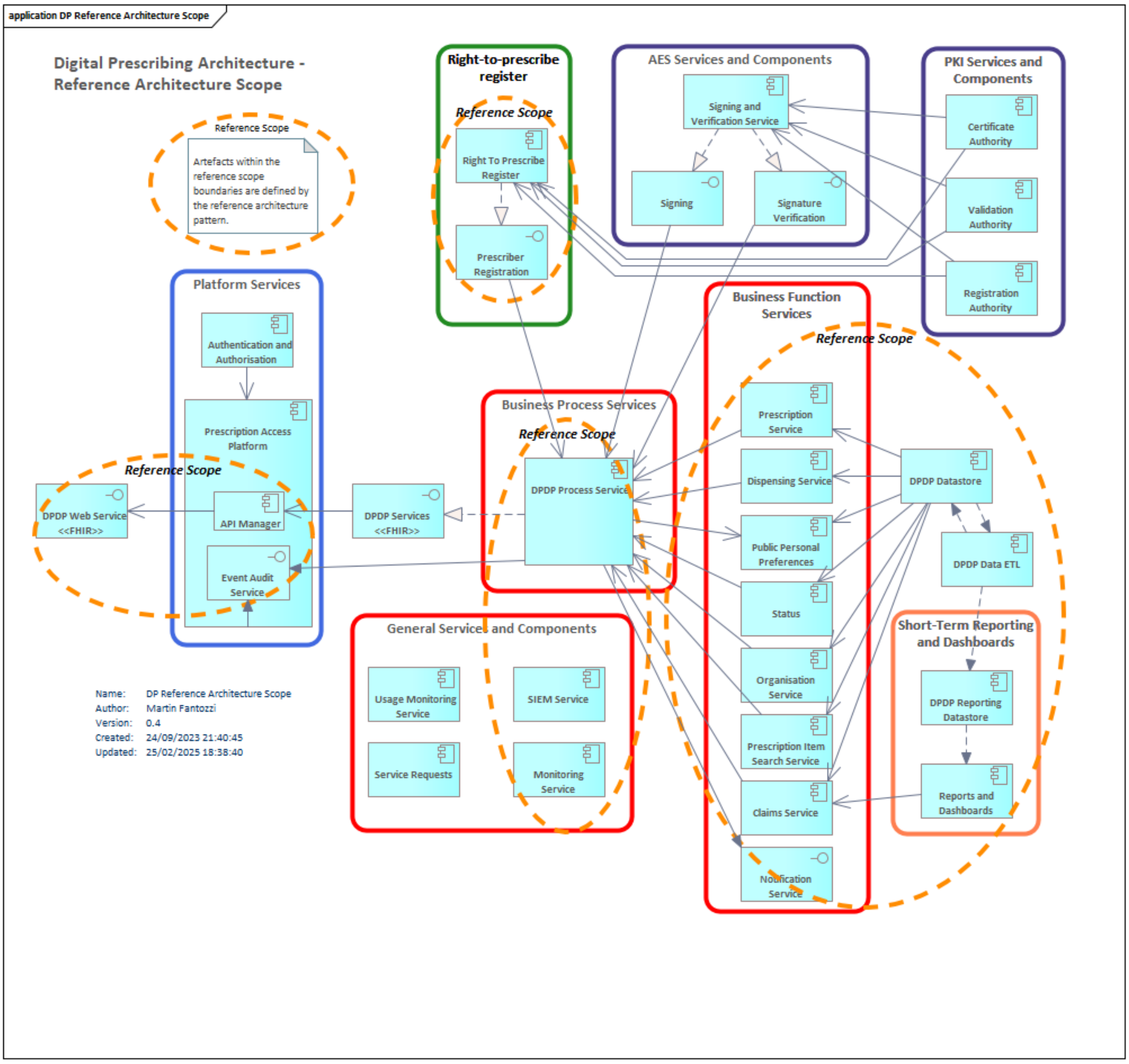
The reference architecture for DPDP contains the key architectural features required to implement the micro-service architecture within an open, cloud-deployed context. Specifically the reference architecture will address the physical architecture for aspects of the Platform Services, Business Process Services, Business Function Services, Right-to-Prescribe Register, General Services and Components and DPDP Datastores defined in [9].

The core characteristics required for DPDP are defined by the architectural principles [7] and the draft non-functional requirements (NFRs) [5]. In general, the service is defined as critical national infrastructure and must both conform to the appropriate security standards and provide a resilient, scalable service at all times. The reference architecture provides core capabilities to address aspects of those NFRs however the detailed design and configuration of the service must ensure that those capabilities are utilised effectively.

Contextual Views of the Reference Architecture

The reference architecture covers several of the DPDP subdomains, as mentioned above. The scope of the reference architecture is intended to cover those subdomain components and services where the programme is either bringing technology into play within NSS DAS or NES Technology Services, or where the intended production usage is outwith the existing use-cases and scenarios for the technology platform within those organisations. With this in mind, some of the referenced platforms may be familiar to the audience but not have been deployed to a production environment in the same usage scenarios.

Reference Architecture Context Diagram


application DP Reference Architecture Scope
Digital Prescribing Architecture - Reference Architecture Scope
Reference Scope
Artefacts within the reference scope boundaries are defined by the reference architecture pattern.
Right-to-prescribe register
Reference Scope
Right To Prescribe Register
Prescriber Registration
AES Services and Components
Signing and Verification Service
Signing
Signature Verification
PKI Services and Components
Certificate Authority
Validation Authority
Registration Authority
Platform Services
Authentication and Authorisation
Prescription Access Platform
Reference Scope
API Manager
Event Audit Service
DPDP Web Service <<FHIR>>
DPDP Services <<FHIR>>
Business Process Services
Reference Scope
DPDP Process Service
Business Function Services
Reference Scope
Prescription Service
Dispensing Service
Public Personal Preferences
Status
Organisation Service
Prescription Item Search Service
Claims Service
Notification Service
DPDP Datastore
DPDP Data ETL
Short-Term Reporting and Dashboards
DPDP Reporting Datastore
Reports and Dashboards
General Services and Components
Usage Monitoring Service
SIEM Service
Service Requests
Monitoring Service
Name: DP Reference Architecture Scope
Author: Martin Fantozzi
Version: 0.4
Created: 24/09/2023 21:40:45
Updated: 25/02/2025 18:38:40

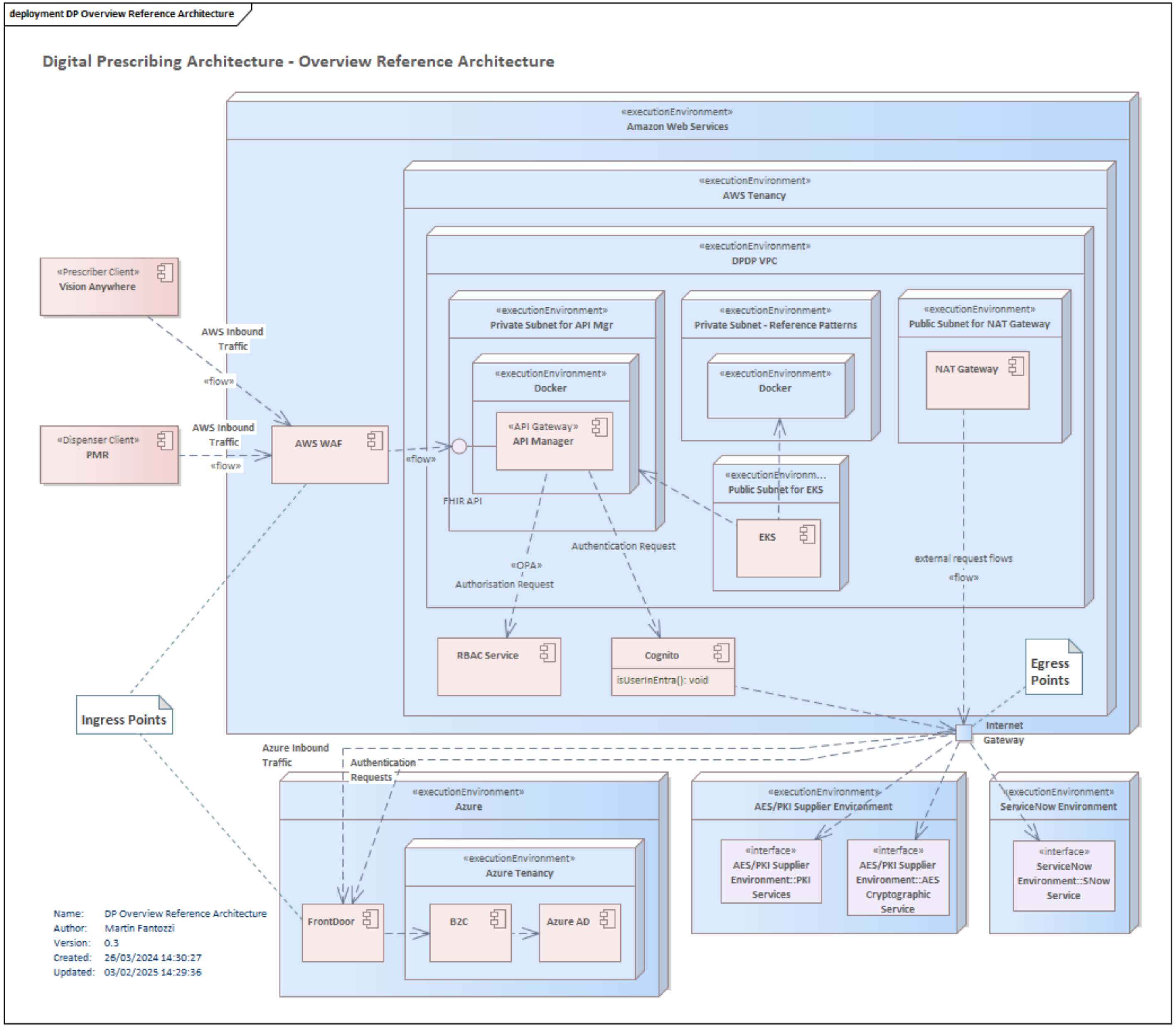
Reference Physical Architecture

The following diagrams shows the physical view of the reference architecture. The initial view contains an abstracted overview of the physical platforms, products and technologies that will be used to build and deploy the subdomains within the reference architecture scope.

The architecture is primarily based on a containerised deployment of micro-services to Amazon AWS, however at a logical level the reference architecture can be deployed to Azure and other cloud platforms.

Reference Physical Architecture - Overview Diagram

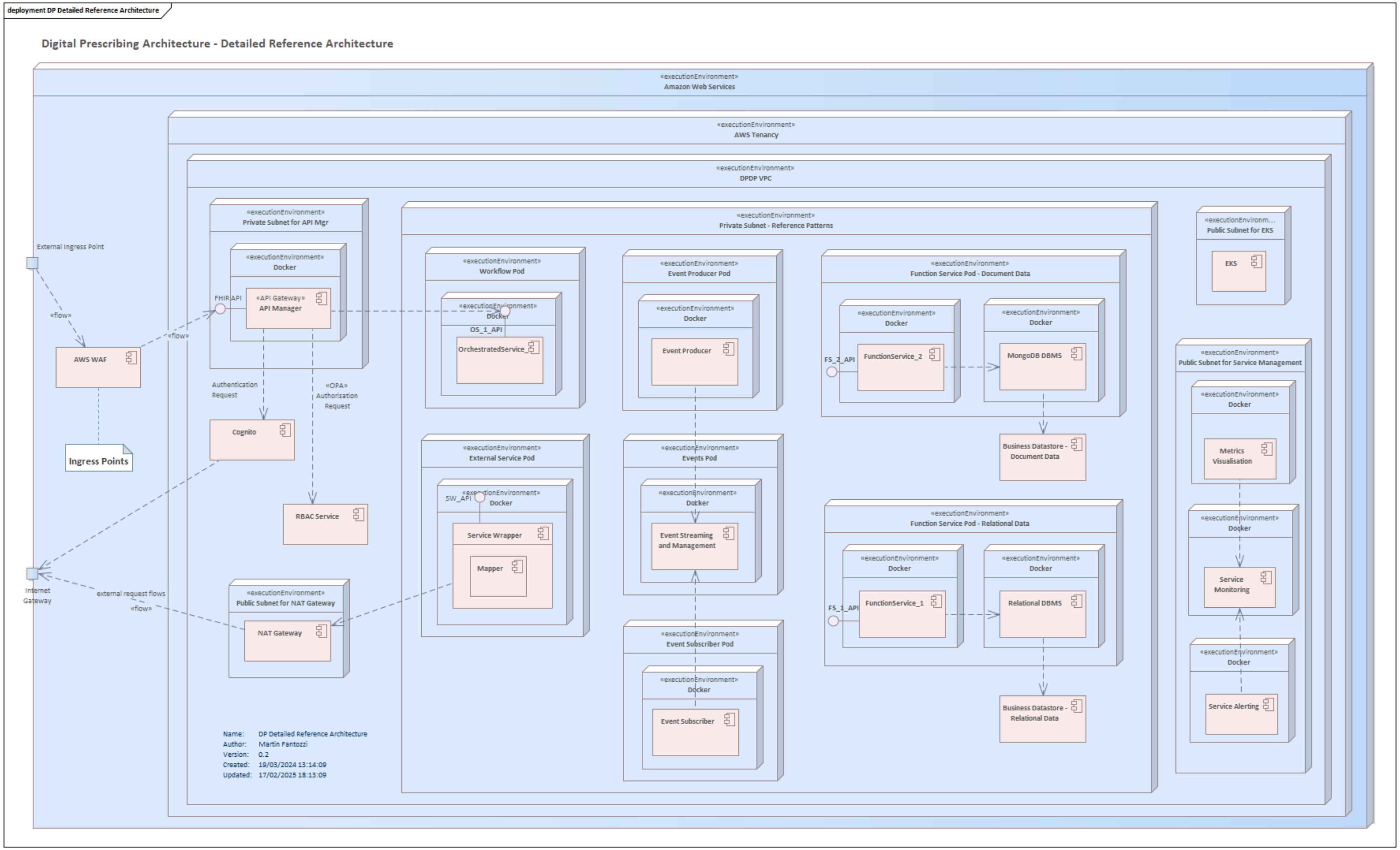
Note that not all dependencies are shown in the overview diagram, for example all Docker instances will be managed by Kubernetes (EKS in the diagram).


deployment DP Overview Reference Architecture
Digital Prescribing Architecture - Overview Reference Architecture
«executionEnvironment» Amazon Web Services
«executionEnvironment» AWS Tenancy
«executionEnvironment» DPDP VPC
«executionEnvironment» Private Subnet for API Mgr
«executionEnvironment» Docker
«API Gateway» API Manager
«executionEnvironment» Private Subnet - Reference Patterns
«executionEnvironment» Docker
«executionEnvironment» Public Subnet for NAT Gateway
NAT Gateway
«executionEnvironm... Public Subnet for EKS
EKS
«Prescriber Client» Vision Anywhere
«Dispenser Client» PMR
AWS Inbound Traffic
«flow»
AWS Inbound Traffic
«flow»
AWS WAF
«flow»
FHIR API
Authentication Request
«OPA» Authorisation Request
external request flows
«flow»
RBAC Service
Cognito
isUserInEntra(): void
Egress Points
Ingress Points
Internet Gateway
Azure Inbound Traffic
Authentication Requests
«executionEnvironment» Azure
«executionEnvironment» Azure Tenancy
FrontDoor
B2C
Azure AD
«executionEnvironment» AES/PKI Supplier Environment
«interface» AES/PKI Supplier Environment::PKI Services
«interface» AES/PKI Supplier Environment::AES Cryptographic Service
«executionEnvironment» ServiceNow Environment
«interface» ServiceNow Environment::SNow Service

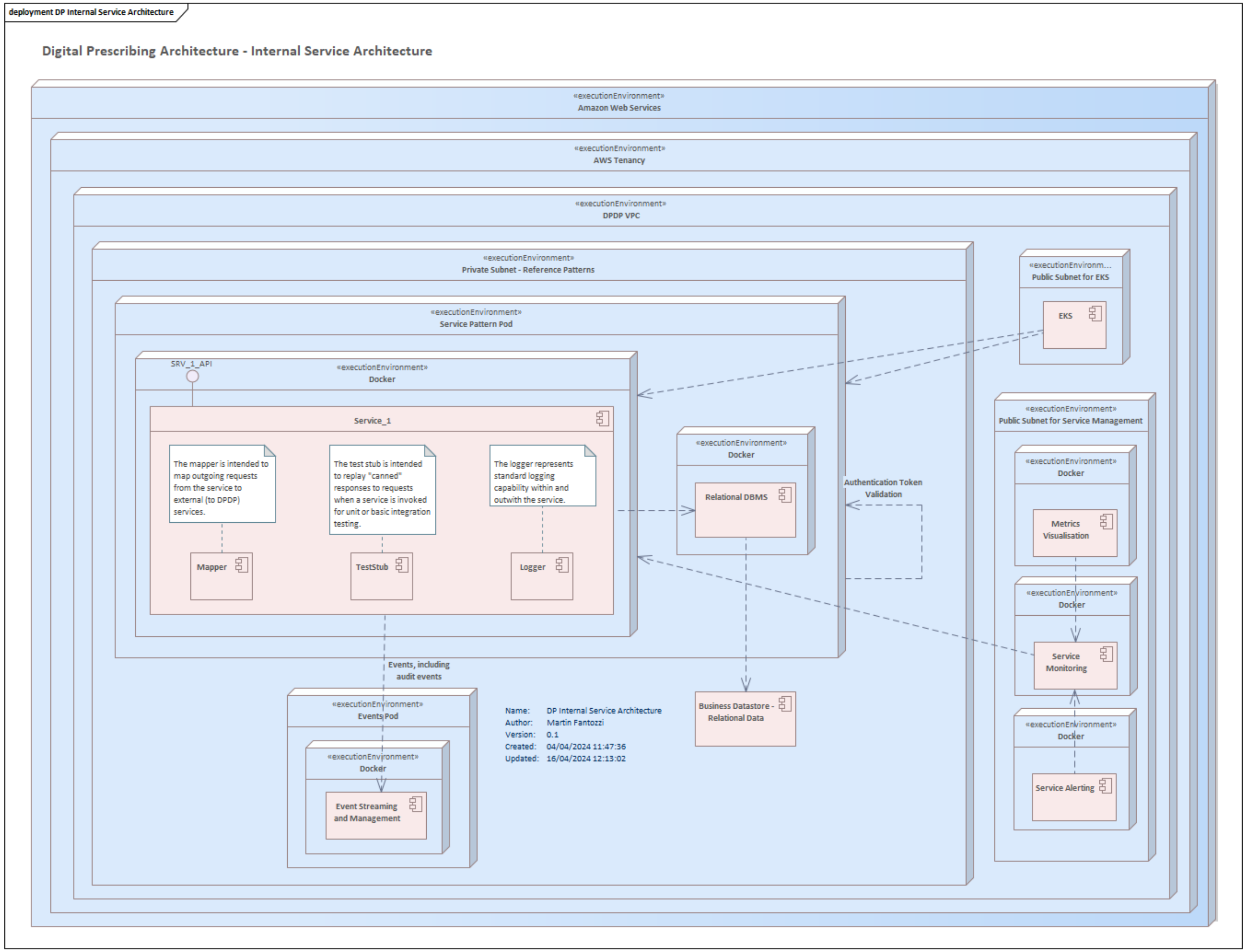
Name:	DP Overview Reference Architecture
Author:	Martin Fantozzi
Version:	0.3
Created:	26/03/2024 14:30:27
Updated:	03/02/2025 14:29:36

Reference Physical Architecture – Detail Diagram

The detailed view of the reference architecture is on the following pages for space reasons.


deployment DP Detailed Reference Architecture
Digital Prescribing Architecture - Detailed Reference Architecture
«executionEnvironment» Amazon Web Services
«executionEnvironment» AWS Tenancy
«executionEnvironment» DPDP VPC
External Ingress Point
«flow»
AWS WAF
«flow»
Ingress Points
«executionEnvironment» Private Subnet for API Mgr
«executionEnvironment» Docker
FHIR API
«API Gateway» API Manager
Authentication Request
«OPA» Authorisation Request
Cognito
RBAC Service
Internet Gateway
external request flows
«flow»
«executionEnvironment» Public Subnet for NAT Gateway
NAT Gateway
«executionEnvironment» Private Subnet - Reference Patterns
«executionEnvironment» Workflow Pod
«executionEnvironment» Docker
OS_1_API
OrchestratedService_
«executionEnvironment» External Service Pod
SW_API
«executionEnvironment» Docker
Service Wrapper
Mapper
«executionEnvironment» Event Producer Pod
«executionEnvironment» Docker
Event Producer
«executionEnvironment» Events Pod
«executionEnvironment» Docker
Event Streaming and Management
«executionEnvironment» Event Subscriber Pod
«executionEnvironment» Docker
Event Subscriber
«executionEnvironment» Function Service Pod - Document Data
«executionEnvironment» Docker
FS_2_API
FunctionService_2
«executionEnvironment» Docker
MongoDB DBMS
Business Datastore - Document Data
«executionEnvironment» Function Service Pod - Relational Data
«executionEnvironment» Docker
FS_1_API
FunctionService_1
«executionEnvironment» Docker
Relational DBMS
Business Datastore - Relational Data
«executionEnvironm... Public Subnet for EKS
EKS
«executionEnvironment» Public Subnet for Service Management
«executionEnvironment» Docker
Metrics Visualisation
«executionEnvironment» Docker
Service Monitoring
«executionEnvironment» Docker
Service Alerting
Name: DP Detailed Reference Architecture
Author: Martin Fantozzi
Version: 0.2
Created: 19/03/2024 13:14:09
Updated: 17/02/2025 18:13:09

Reference Physical Architecture – Internal Pod and Service Architecture



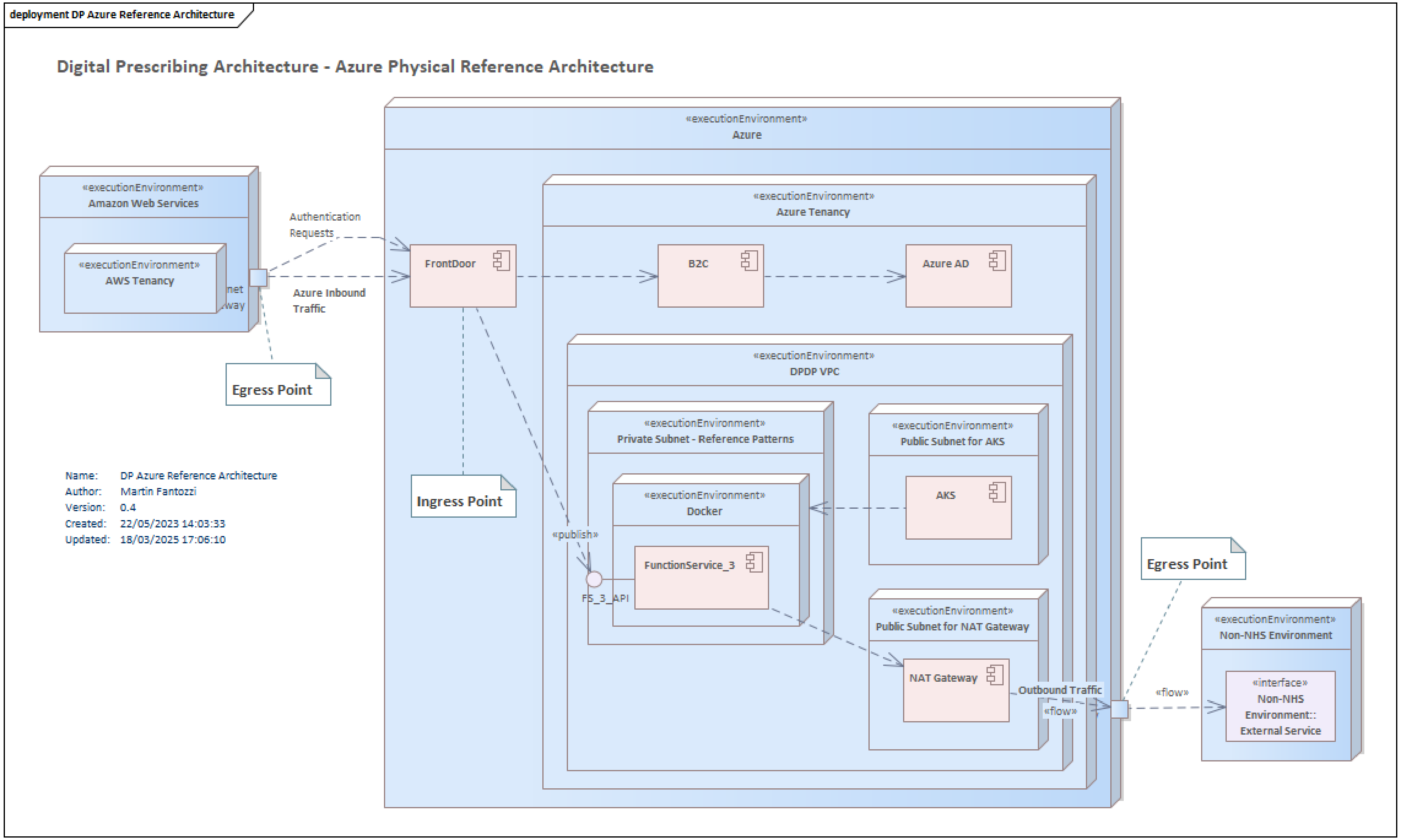
The mapper is intended to map outgoing requests from the service to external (to DPDP) services.

The test stub is intended to replay "canned" responses to requests when a service is invoked for unit or basic integration testing.

The logger represents standard logging capability within and outwith the service.

Name: DP Internal Service Architecture
Author: Martin Fantozzi
Version: 0.1
Created: 04/04/2024 11:47:36
Updated: 16/04/2024 12:13:02

Reference Physical Architecture – Azure Reference Architecture



Reference Architecture Technologies and Platforms

The following table gives a brief outline of the technology and platforms for specific services and components within the reference architecture.

Subdomain (if applicable)	Component	Technology / Platform	Notes
Platform Services	API Manager	(TBD alongside NES NDP team)	This will be either a COTS offering or an OS offering bundled with enterprise support from a well-established vendor
Cloud tenancy boundaries	Web Application Firewall / API Gateway	Amazon AWS WAF, Microsoft FrontDoor	Basic access control policy enforcement for cloud-to-cloud service invocation
Authorisation and control between containers	All DPDP-created micro-services	Container management service (see below)	Self-encoded tokens will be authenticated by each container.
Business Process Service	Orchestration	Docker-deployed C#	There will be a single micro-service for each defined API request.
All DPDP services	Deployment container for business processes, business functions, auditing and monitoring services, DBMSs, API manager, Right-to-Prescribe, etc	Docker	The containers will contain a single micro service or platform deployment each.
Deployment environments	Container management service	Kubernetes wrapped as AWS AKS	This will be a wrapped offering from the cloud provider, specifically AKS for AWS. EKS will be used in the future for any Azure-deployed micro-services.
Business Functions	Mapping layer pattern. Used where applicable	C#/JSON/XML	Mapping pattern internal to business functions to map from common data model to specific local domain if applicable, such as for sending data to an external notification service.
Event output from all micro-services	Event streaming service	TBC (possibly Apache Kafka)	The event streamer is an underlying integration technology enabling complex asynchronous event processing

Subdomain (if applicable)	Component	Technology / Platform	Notes
Platform, Supporting Services	Audit service, including audit datastore	NDP Audit Service, using MongoDB datastore	Populated via audit event stream. Work ongoing to refine both the service and the target data persistence platform
Platform, Supporting Services	Service monitoring	TBC	Ongoing evaluation
Platform, Supporting Services	SIEM monitoring	NSS SOC	Assumption is that this will be fed from NDP auditing service, other event streams and possibly log scraping.
Platform, Supporting Services	Appropriate use monitoring	Imprivata Fair Warning	Will be fed from extract files created within the NDP audit service
Platform, Supporting Services	Service Request service	ServiceNow, aka SNow (***to be replaced possibly late 2025***)	Direct integration with RtP to allow programmatic raising of service requests for prescriber Entra account updates
Short-term Reporting	Short-term Reporting / Datastore	PowerBI tool and Seer datastore in Azure	To be used to monitor rollout activity and progress; Fed from event streams
DPDP Datastore	Transactional message data, audit data	MongoDB	There will be a mix of data storage technologies dependent on the data requirements of the controlling service.

Additional Technology Dependencies

The following table includes brief descriptions of key technologies upon which the reference architecture has dependencies. The detailed physical architectures of the subdomains are defined in the relevant subsection.

Subdomain	Component	Technology / Platform	Notes
DPDP Datastore	Reference and highly structured data	RDBMS TBC	There will be a mix of data storage technologies dependent on the data requirements of the controlling service.
Platform Services	Authorisation Service	TBC but probably OPA-compliant service. Invoked from API Manager	RBAC conforming to OPA

Subdomain	Component	Technology / Platform	Notes
Platform Services	Authentication and Authorisation - IAM		Specifically the centralised M365 AAD deployment managed by NSS DAS. This will be integrated with the NDP platform using AWS Cognito to access the centralised AAD directory service.
Additional Authentication	Multi-Factor Authentication (MFA)	Microsoft Entra (TBC)	

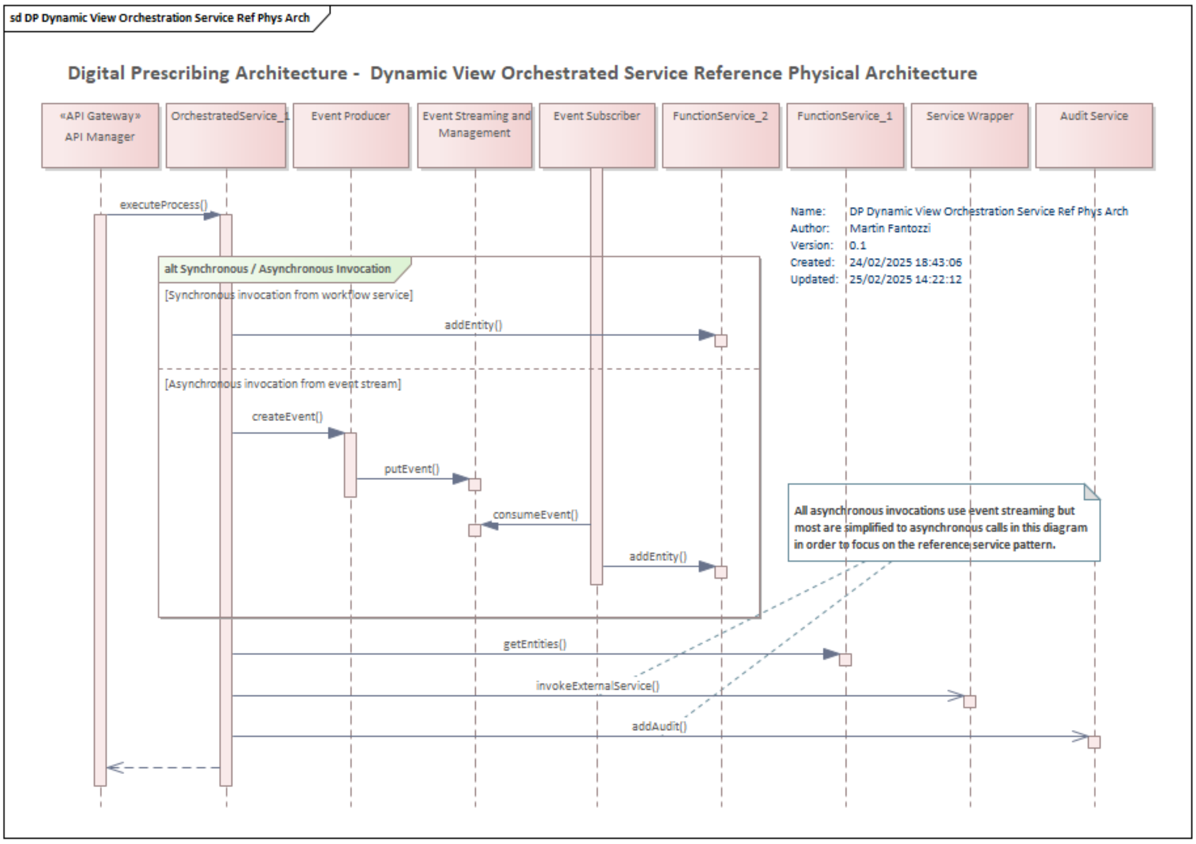
Dynamic Physical Reference Architecture – Service and Collaboration Behaviour

The dynamic architecture refers to the collaborative behaviour of the services, platforms and external dependencies in order to realise the objectives of a specific service invocation. The following diagrams refer to the intended behaviour of services that conform to the reference architecture, and uses the example services of that architecture.

The diagrams are presented as sequence diagrams to show the order of invocation, and includes examples of both synchronous and asynchronous invocation. The latter follows an event-driven architecture paradigm and uses the event-streaming platform within the overall physical architecture, however in order to keep the diagrams clear only the first asynchronous invocation in each diagram shows the full set of event interactions, and subsequent asynchronous behaviour is indicated using standard UML sequence diagram notation of an open-ended arrow.

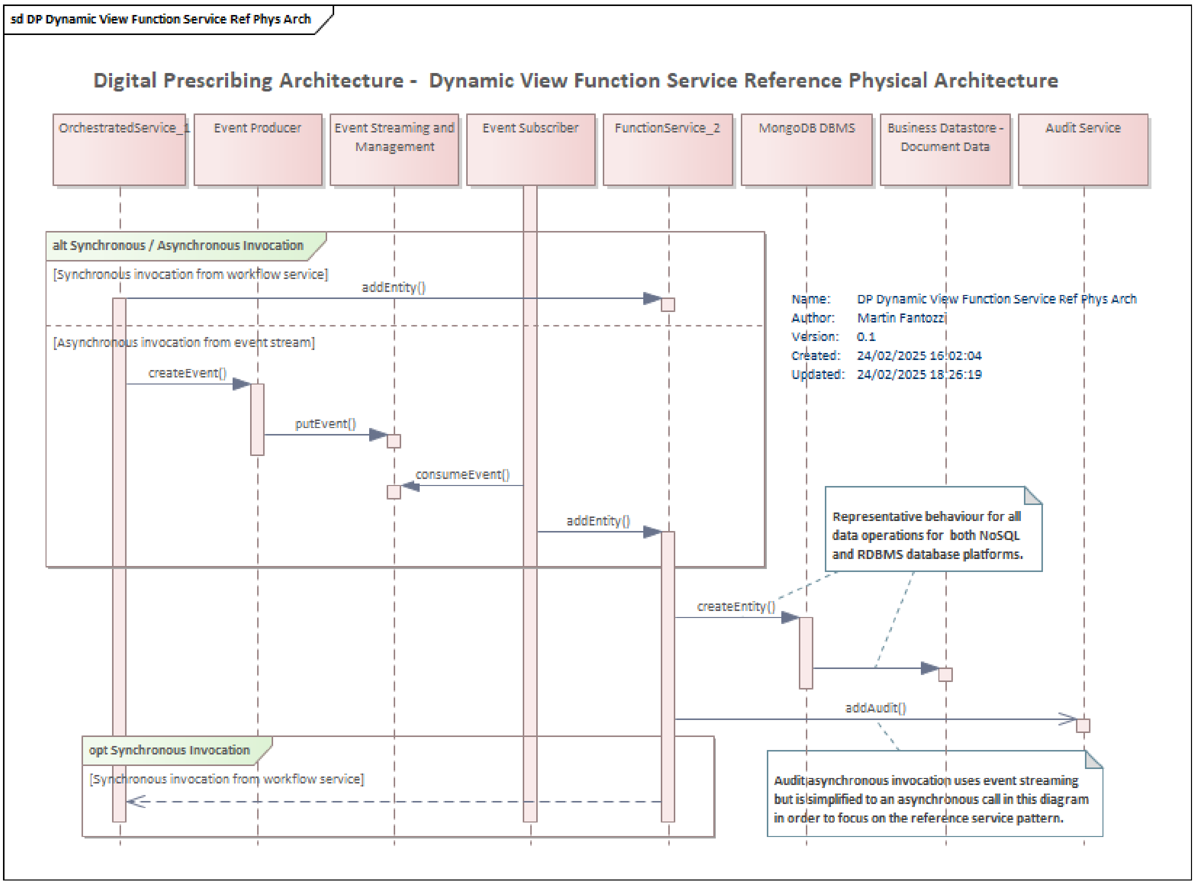
Orchestration Service Collaboration

The orchestration service collaboration diagram is focussed on the responsibilities of the orchestrated and choreographed workflow services, showing invocation by the API manager and use of the underlying function and external services. All workflow services will be invoked synchronously. All services will generate audit events that are passed to the audit service using the event streaming platform.


sd DP Dynamic View Orchestration Service Ref Phys Arch
Digital Prescribing Architecture - Dynamic View Orchestrated Service Reference Physical Architecture
«API Gateway» API Manager
OrchestratedService_1
Event Producer
Event Streaming and Management
Event Subscriber
FunctionService_2
FunctionService_1
Service Wrapper
Audit Service
executeProcess()
Name: DP Dynamic View Orchestration Service Ref Phys Arch
Author: Martin Fantozzi
Version: 0.1
Created: 24/02/2025 18:43:06
Updated: 25/02/2025 14:22:12
alt Synchronous / Asynchronous Invocation
[Synchronous invocation from workflow service]
addEntity()
[Asynchronous invocation from event stream]
createEvent()
putEvent()
consumeEvent()
addEntity()
All asynchronous invocations use event streaming but most are simplified to asynchronous calls in this diagram in order to focus on the reference service pattern.
getEntities()
invokeExternalService()
addAudit()

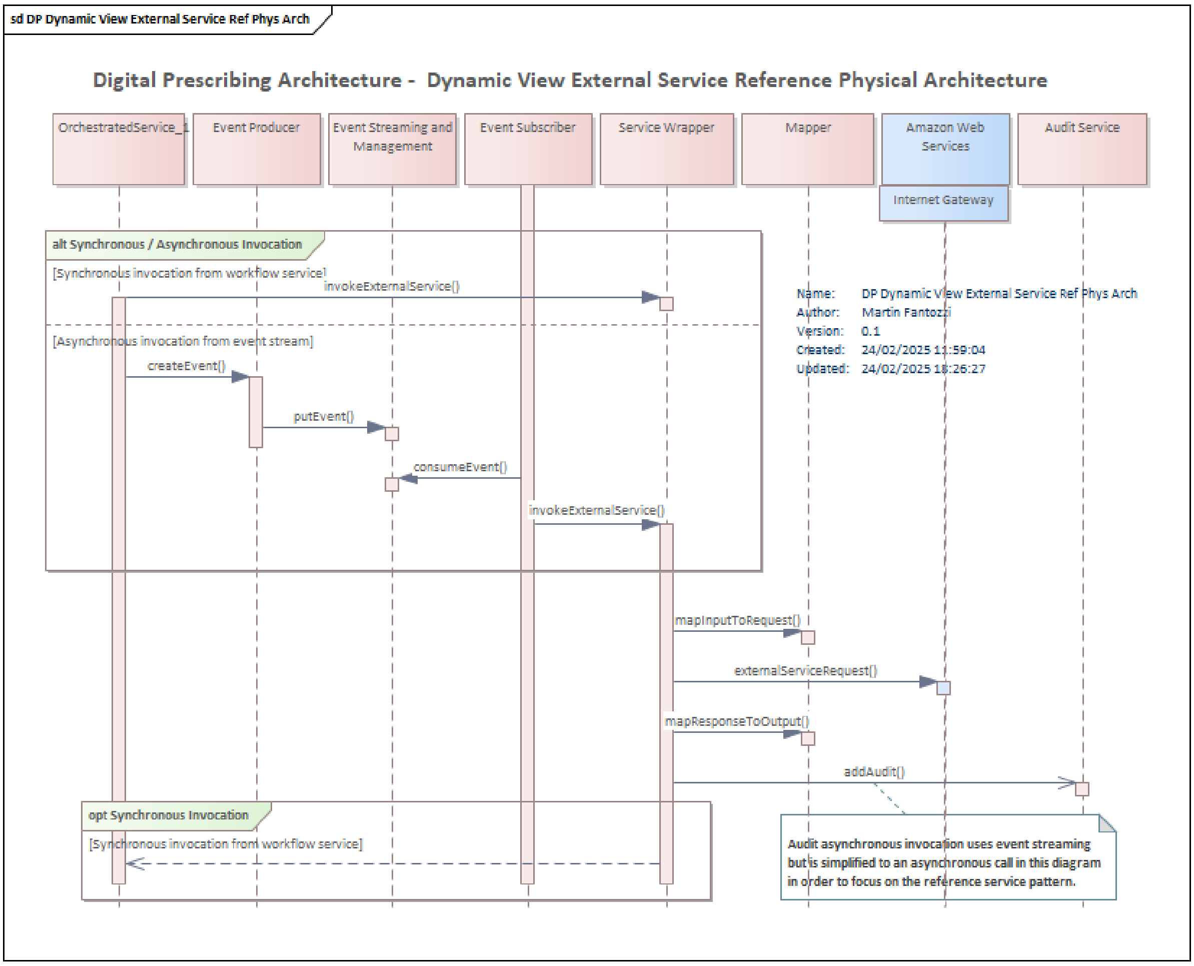
Function Service

The business function service collaboration diagram is focussed on the responsibilities of a function service, showing both synchronous and asynchronous invocations of that service. In the following diagram it accesses a NoSQL database however both relational and document databases will be deployed within the context of DPDP. All services will generate audit events that are passed to the audit service using the event streaming platform.


sd DP Dynamic View Function Service Ref Phys Arch
Digital Prescribing Architecture - Dynamic View Function Service Reference Physical Architecture
OrchestratedService_1
Event Producer
Event Streaming and Management
Event Subscriber
FunctionService_2
MongoDB DBMS
Business Datastore - Document Data
Audit Service
alt Synchronous / Asynchronous Invocation
[Synchronous invocation from workflow service]
addEntity()
[Asynchronous invocation from event stream]
createEvent()
putEvent()
consumeEvent()
addEntity()
createEntity()
addAudit()
opt Synchronous Invocation
[Synchronous invocation from workflow service]
Name: DP Dynamic View Function Service Ref Phys Arch
Author: Martin Fantozzi
Version: 0.1
Created: 24/02/2025 16:02:04
Updated: 24/02/2025 18:26:19
Representative behaviour for all data operations for both NoSQL and RDBMS database platforms.
Audit asynchronous invocation uses event streaming but is simplified to an asynchronous call in this diagram in order to focus on the reference service pattern.

External Service Collaboration

The external service collaboration diagram is focussed on the responsibilities of a facade service accessing an external service or system, showing both synchronous and asynchronous invocations of that service. DPDP will integrate with a small number of critical external services, including but not limited to AES and PKI services, ePMS and short-term reporting. The following diagram shows the use of an internal mapper component to map from the common data model used across DPDP to the data format and protocols in use by the external service. All services will generate audit events that are passed to the audit service using the event streaming platform.



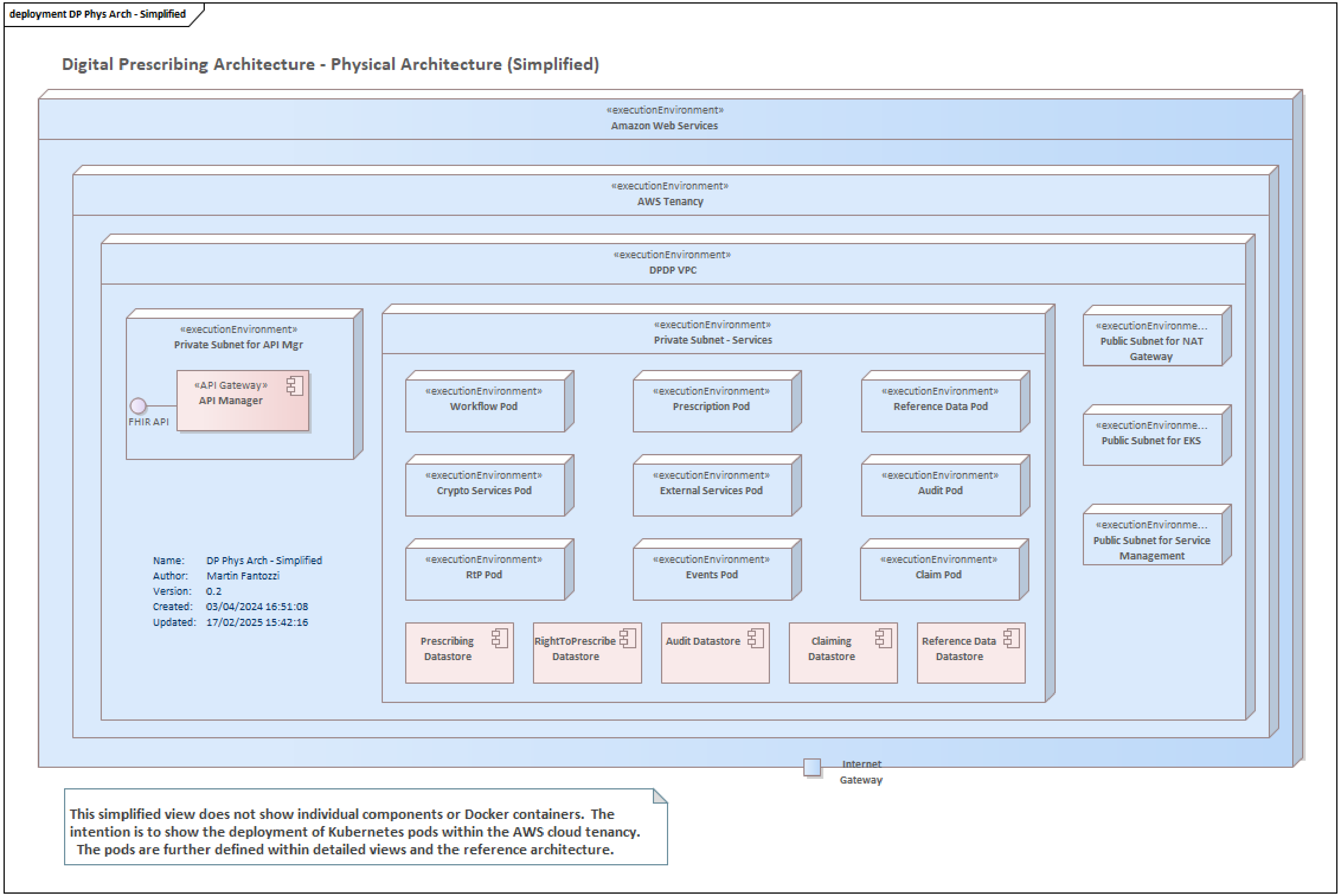
sd DP Dynamic View External Service Ref Phys Arch
Digital Prescribing Architecture - Dynamic View External Service Reference Physical Architecture
OrchestratedService_1
Event Producer
Event Streaming and Management
Event Subscriber
Service Wrapper
Mapper
Amazon Web Services
Internet Gateway
Audit Service
alt Synchronous / Asynchronous Invocation
[Synchronous invocation from workflow service]
invokeExternalService()
[Asynchronous invocation from event stream]
createEvent()
putEvent()
consumeEvent()
invokeExternalService()
mapInputToRequest()
externalServiceRequest()
mapResponseToOutput()
addAudit()
opt Synchronous Invocation
[Synchronous invocation from workflow service]
Name: DP Dynamic View External Service Ref Phys Arch
Author: Martin Fantozzi
Version: 0.1
Created: 24/02/2025 11:59:04
Updated: 24/02/2025 18:26:27
Audit asynchronous invocation uses event streaming but is simplified to an asynchronous call in this diagram in order to focus on the reference service pattern.

Physical Architecture Overview

The following views provide:

- a contextual view of the physical architecture in terms of the deployed pods of DPDP containers within the AWS cloud tenancy and the supporting cloud and external platforms that support this deployment;
- a simplified representation of the key characteristics of the service pods as realisations of the pod patterns represented in the Reference Physical Architecture – Internal Pod and Service Architecture view;
- external dependencies for the overall DPDP service, such as externally-sourced products and services deployed to vendor environments outside of the existing AWS tenancy.

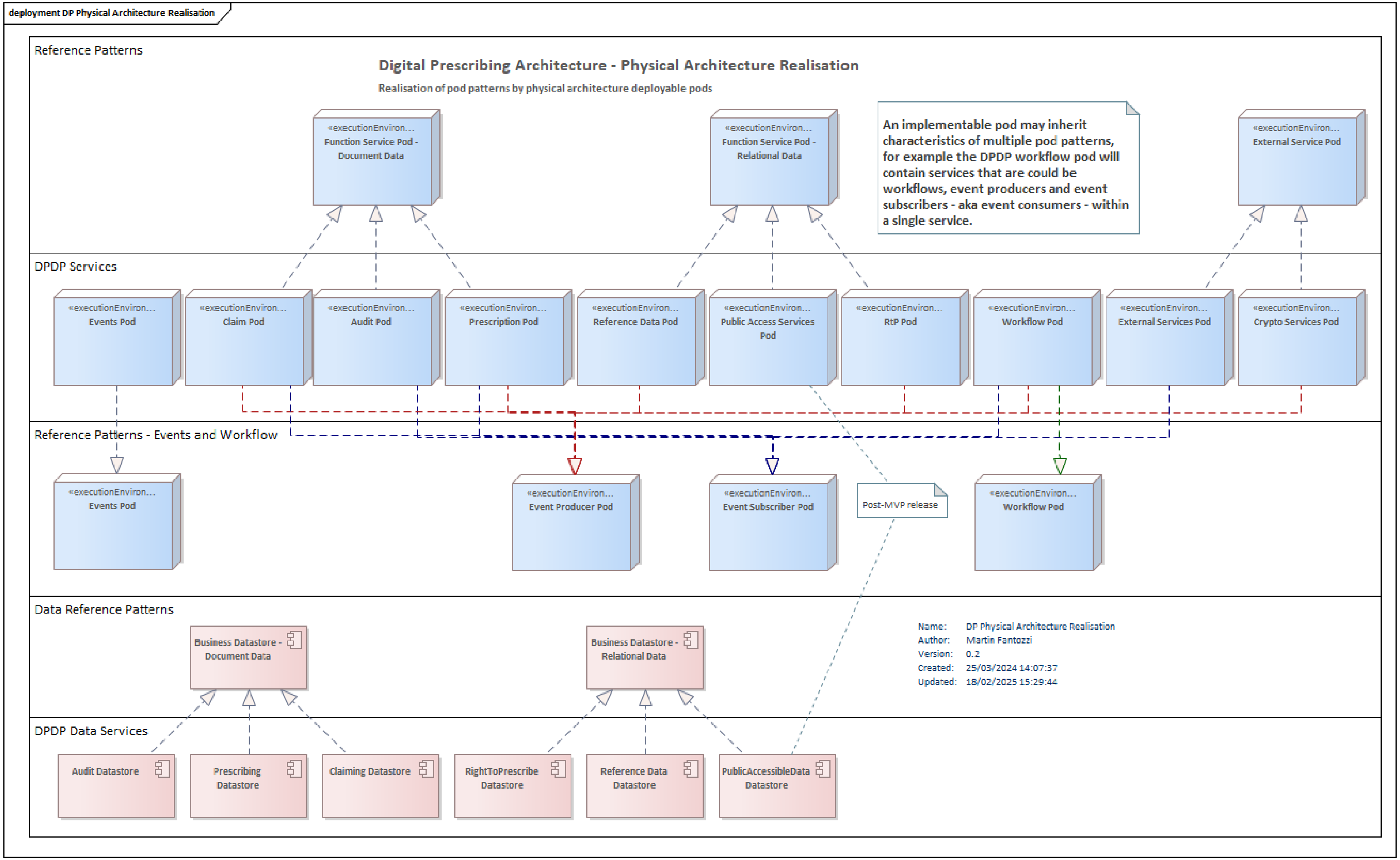
Physical Architecture – Overall Simplified View


deployment DP Phys Arch - Simplified
Digital Prescribing Architecture - Physical Architecture (Simplified)
«executionEnvironment» Amazon Web Services
«executionEnvironment» AWS Tenancy
«executionEnvironment» DPDP VPC
«executionEnvironment» Private Subnet for API Mgr
«API Gateway» API Manager
FHIR API
«executionEnvironment» Private Subnet - Services
«executionEnvironment» Workflow Pod
«executionEnvironment» Prescription Pod
«executionEnvironment» Reference Data Pod
«executionEnvironment» Crypto Services Pod
«executionEnvironment» External Services Pod
«executionEnvironment» Audit Pod
«executionEnvironment» RtP Pod
«executionEnvironment» Events Pod
«executionEnvironment» Claim Pod
Prescribing Datastore
RightToPrescribe Datastore
Audit Datastore
Claiming Datastore
Reference Data Datastore
«executionEnvironme... Public Subnet for NAT Gateway
«executionEnvironme... Public Subnet for EKS
«executionEnvironme... Public Subnet for Service Management
Internet Gateway

Name:	DP Phys Arch - Simplified
Author:	Martin Fantozzi
Version:	0.2
Created:	03/04/2024 16:51:08
Updated:	17/02/2025 15:42:16

This simplified view does not show individual components or Docker containers. The intention is to show the deployment of Kubernetes pods within the AWS cloud tenancy. The pods are further defined within detailed views and the reference architecture.

Physical Architecture Realisations



Digital Prescribing Architecture - Physical Architecture Realisation

Realisation of pod patterns by physical architecture deployable pods

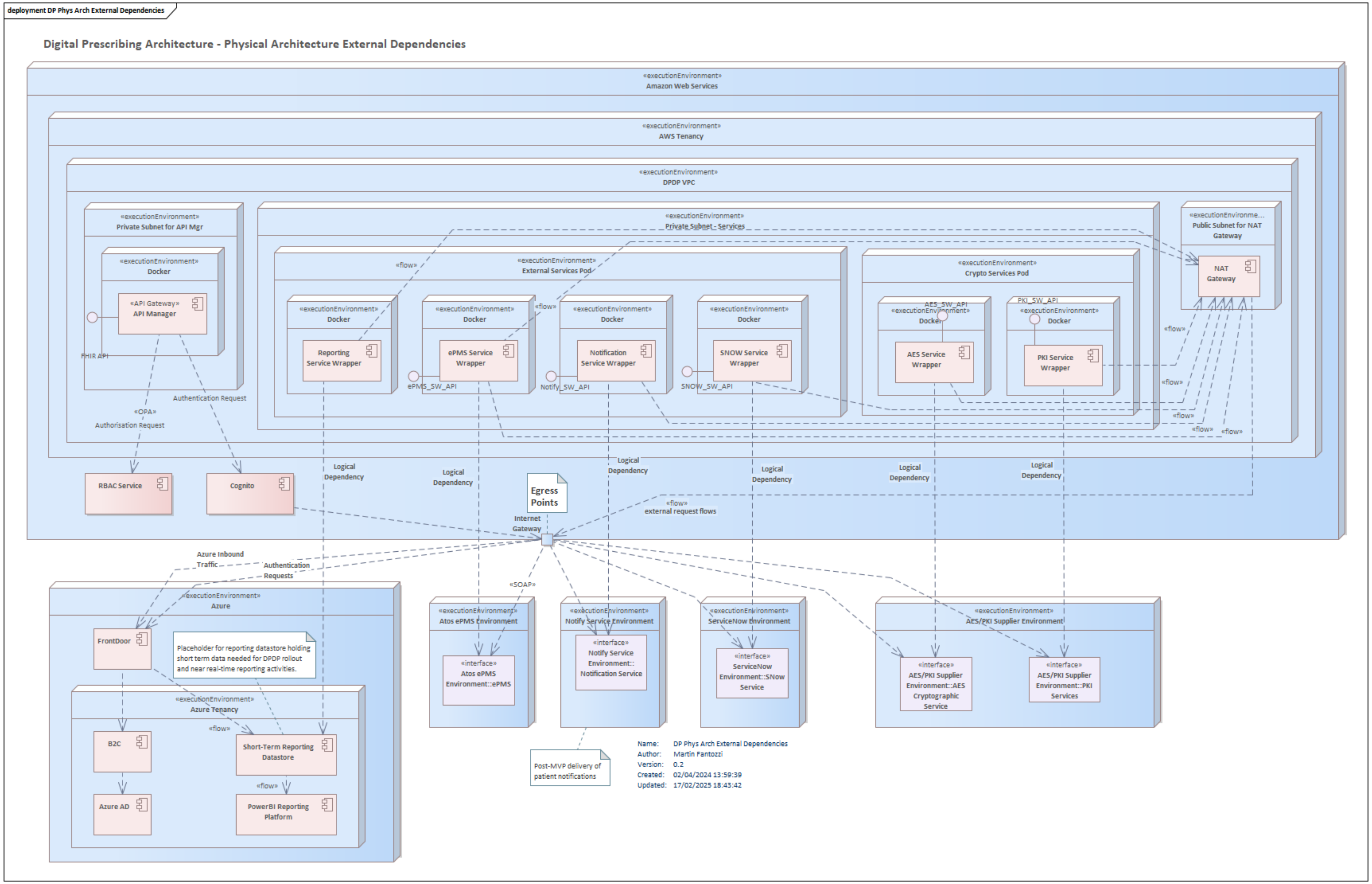
An implementable pod may inherit characteristics of multiple pod patterns, for example the DPDP workflow pod will contain services that are could be workflows, event producers and event subscribers - aka event consumers - within a single service.

Post-MVP release

Name: DP Physical Architecture Realisation
Author: Martin Fantozzi
Version: 0.2
Created: 25/03/2024 14:07:37
Updated: 18/02/2025 15:29:44

deployment DP Physical Architecture Realisation
Reference Patterns
«executionEnviron... Function Service Pod - Document Data
«executionEnviron... Function Service Pod - Relational Data
«executionEnviron... External Service Pod
DPDP Services
«executionEnviron... Events Pod
«executionEnviron... Claim Pod
«executionEnviron... Audit Pod
«executionEnviron... Prescription Pod
«executionEnviron... Reference Data Pod
«executionEnviron... Public Access Services Pod
«executionEnviron... RtP Pod
«executionEnviron... Workflow Pod
«executionEnviron... External Services Pod
«executionEnviron... Crypto Services Pod
Reference Patterns - Events and Workflow
«executionEnviron... Events Pod
«executionEnviron... Event Producer Pod
«executionEnviron... Event Subscriber Pod
«executionEnviron... Workflow Pod
Data Reference Patterns
Business Datastore - Document Data
Business Datastore - Relational Data
DPDP Data Services
Audit Datastore
Prescribing Datastore
Claiming Datastore
RightToPrescribe Datastore
Reference Data Datastore
PublicAccessibleData Datastore

Physical Architecture External Dependencies


deployment DP Phys Arch External Dependencies
Digital Prescribing Architecture - Physical Architecture External Dependencies
«executionEnvironment» Amazon Web Services
«executionEnvironment» AWS Tenancy
«executionEnvironment» DPDP VPC
«executionEnvironment» Private Subnet for API Mgr
«executionEnvironment» Docker
«API Gateway» API Manager
FHIR API
«executionEnvironment» Private Subnet - Services
«executionEnvironment» External Services Pod
«flow»
«executionEnvironment» Docker
Reporting Service Wrapper
«executionEnvironment» Docker
ePMS Service Wrapper
ePMS_SW_API
«flow»
«executionEnvironment» Docker
Notification Service Wrapper
Notify_SW_API
«executionEnvironment» Docker
SNOW Service Wrapper
SNOW_SW_API
«executionEnvironment» Crypto Services Pod
AES_SW_API
«executionEnvironment» Docker
AES Service Wrapper
PKI_SW_API
«executionEnvironment» Docker
PKI Service Wrapper
«executionEnvironme... Public Subnet for NAT Gateway
NAT Gateway
«flow»
«flow»
«flow»
«flow»
«flow»
Authentication Request
«OPA» Authorisation Request
RBAC Service
Cognito
Logical Dependency
Logical Dependency
Logical Dependency
Logical Dependency
Logical Dependency
Logical Dependency
Egress Points
Internet Gateway
«flow» external request flows
Azure Inbound Traffic
Authentication Requests
«SOAP»
«executionEnvironment» Azure
FrontDoor
Placeholder for reporting datastore holding short term data needed for DPDP rollout and near real-time reporting activities.
«executionEnvironment» Azure Tenancy
«flow»
B2C
Short-Term Reporting Datastore
«flow»
Azure AD
PowerBI Reporting Platform
«executionEnvironment» Atos ePMS Environment
«interface» Atos ePMS Environment::ePMS
«executionEnvironment» Notify Service Environment
«interface» Notify Service Environment:: Notification Service
Post-MVP delivery of patient notifications
«executionEnvironment» ServiceNow Environment
«interface» ServiceNow Environment::SNow Service
«executionEnvironment» AES/PKI Supplier Environment
«interface» AES/PKI Supplier Environment::AES Cryptographic Service
«interface» AES/PKI Supplier Environment::PKI Services
Name: DP Phys Arch External Dependencies
Author: Martin Fantozzi
Version: 0.2
Created: 02/04/2024 13:59:39
Updated: 17/02/2025 18:43:42

Platform Services Physical Architecture

The platform services provide the common endpoint for all access to the DPDP services. They provide a set of URLs for each published environment, along with direct integration to a simple authentication and authorisation service.

The physical architecture of the platform services is primarily defined by the architectural design of the National Digital Platform, however the audit service and API management platform are both described within the reference architecture.

Architecture Diagram

Component Descriptions

Prescription Access Platform

API Manager

URL Redirect

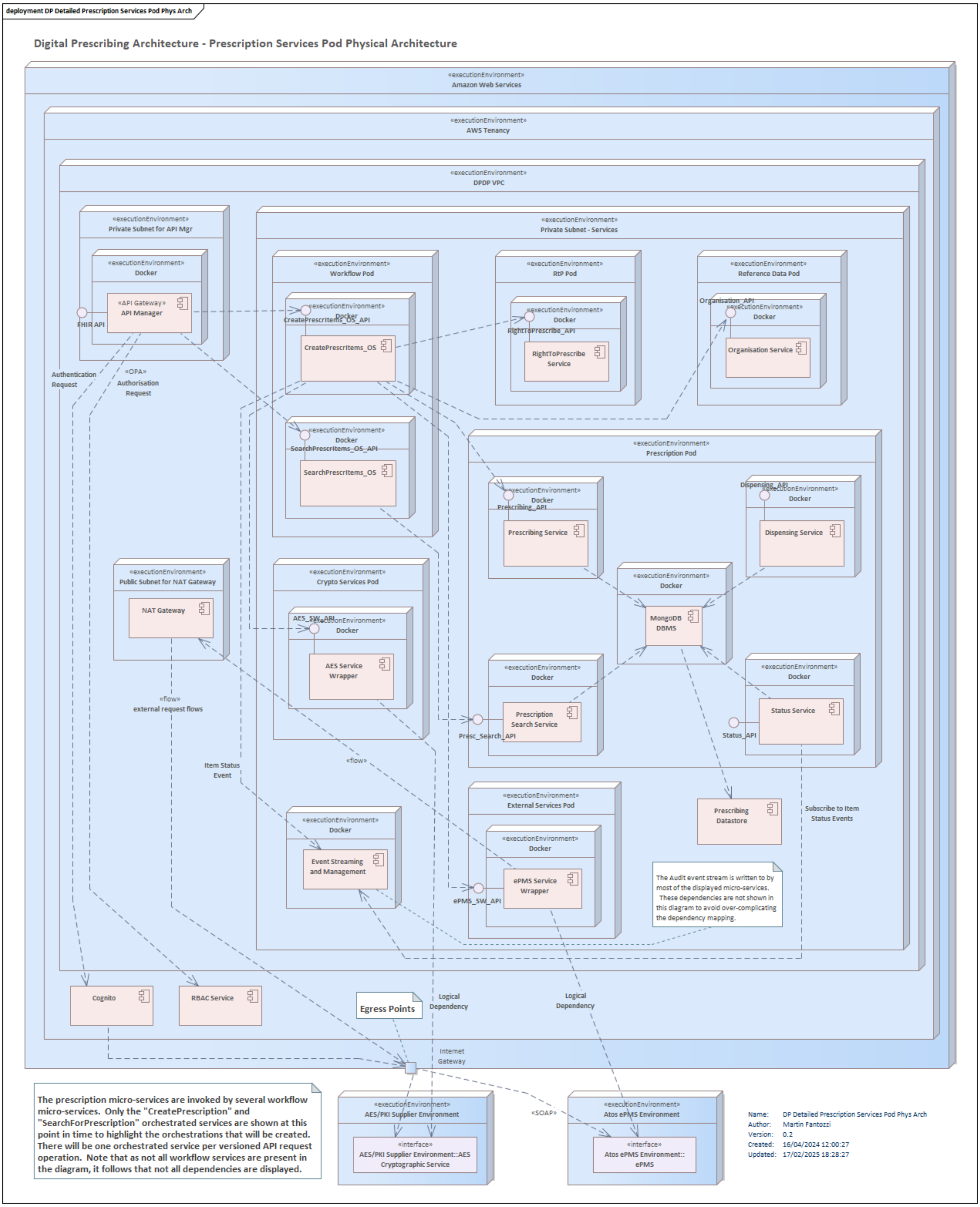
Audit Service

Integration with Amazon AWS Cognito and AAD

Prescription Services and Related Pods – Physical Architecture

The Prescription Services pod and dependencies highlight the physical architecture of most of the key functional areas within the overall DPDP service. The following diagram shows not only the internal services deployed within their container context, but also the business process pod with appropriate example workflow services and a number of other depended-upon services deployed within their own containers and pods.

Architecture Diagram


deployment DP Detailed Prescription Services Pod Phys Arch
Digital Prescribing Architecture - Prescription Services Pod Physical Architecture
«executionEnvironment» Amazon Web Services
«executionEnvironment» AWS Tenancy
«executionEnvironment» DPDP VPC
«executionEnvironment» Private Subnet for API Mgr
«executionEnvironment» Docker
«API Gateway» API Manager
FHIR API
Authentication Request
«OPA» Authorisation Request
«executionEnvironment» Private Subnet - Services
«executionEnvironment» Workflow Pod
«executionEnvironment» Docker
CreatePrescrItems_OS_API
CreatePrescrItems_OS
«executionEnvironment» Docker
SearchPrescrItems_OS_API
SearchPrescrItems_OS
«executionEnvironment» RtP Pod
«executionEnvironment» Docker
RightToPrescribe_API
RightToPrescribe Service
«executionEnvironment» Reference Data Pod
Organisation_API
«executionEnvironment» Docker
Organisation Service
«executionEnvironment» Prescription Pod
«executionEnvironment» Docker
Prescribing_API
Prescribing Service
Dispensing_API
«executionEnvironment» Docker
Dispensing Service
«executionEnvironment» Docker
MongoDB DBMS
«executionEnvironment» Docker
Prescription Search Service
Presc_Search_API
«executionEnvironment» Docker
Status Service
Status_API
«executionEnvironment» Public Subnet for NAT Gateway
NAT Gateway
«flow» external request flows
«executionEnvironment» Crypto Services Pod
AES_SW_API
«executionEnvironment» Docker
AES Service Wrapper
Item Status Event
«flow»
«executionEnvironment» Docker
Event Streaming and Management
«executionEnvironment» External Services Pod
«executionEnvironment» Docker
ePMS Service Wrapper
ePMS_SW_API
Prescribing Datastore
Subscribe to Item Status Events
Cognito
RBAC Service
Egress Points
Logical Dependency
Logical Dependency
Internet Gateway
«executionEnvironment» AES/PKI Supplier Environment
«interface» AES/PKI Supplier Environment::AES Cryptographic Service
«SOAP»
«executionEnvironment» Atos ePMS Environment
«interface» Atos ePMS Environment:: ePMS

The Audit event stream is written to by most of the displayed micro-services. These dependencies are not shown in this diagram to avoid over-complicating the dependency mapping.

The prescription micro-services are invoked by several workflow micro-services. Only the "CreatePrescription" and "SearchForPrescription" orchestrated services are shown at this point in time to highlight the orchestrations that will be created. There will be one orchestrated service per versioned API request operation. Note that as not all workflow services are present in the diagram, it follows that not all dependencies are displayed.

Name:	DP Detailed Prescription Services Pod Phys Arch
Author:	Martin Fantozzi
Version:	0.2
Created:	16/04/2024 12:00:27
Updated:	17/02/2025 18:28:27

Pod Descriptions

Prescription Services Pod Physical Architecture

Component Descriptions

Prescribing Service

The prescribing service implements the read, creation and update capabilities for prescribed items. These items are a hierarchical set described in further detail in [10].

Dispensing Service

The prescribing service implements the creation and update capabilities for dispensed items. These items are a hierarchical set described in further detail in [10].

Status Service

The status service contains the status-to-status transition rules, which are in turn based on the state machines defined in the State machines section within the description of the Data Architecture [10]. This service is intended to be invoked synchronously to check allowed status transitions, such as ensuring that a cancelled, prescribed item cannot be transferred to a CP, and asynchronously as part of the update process for the status of an item and the calculation of the higher level status of a parent item, for example when a single dispensed item is set to collected the overall prescription is set to partially collected if there are further items that have not yet been picked up by or delivered to a patient.

Prescription Search

The prescription search service provides prescribed and dispensed item search capabilities using different combinations of search criteria. All search results will be returned as a full set of data for the item, including medication information, or partial sets of data where the search has been invoked by a business actor in a role that does not allow direct access to confidential patient information.

MongoDB DBMS and Prescription Datastore

The prescription services use the MongoDB data persistence platform. This will be deployed in its own Docker container within the prescription services pod. The datastore will exist outside of the managed containers.

Business Process Physical Architecture

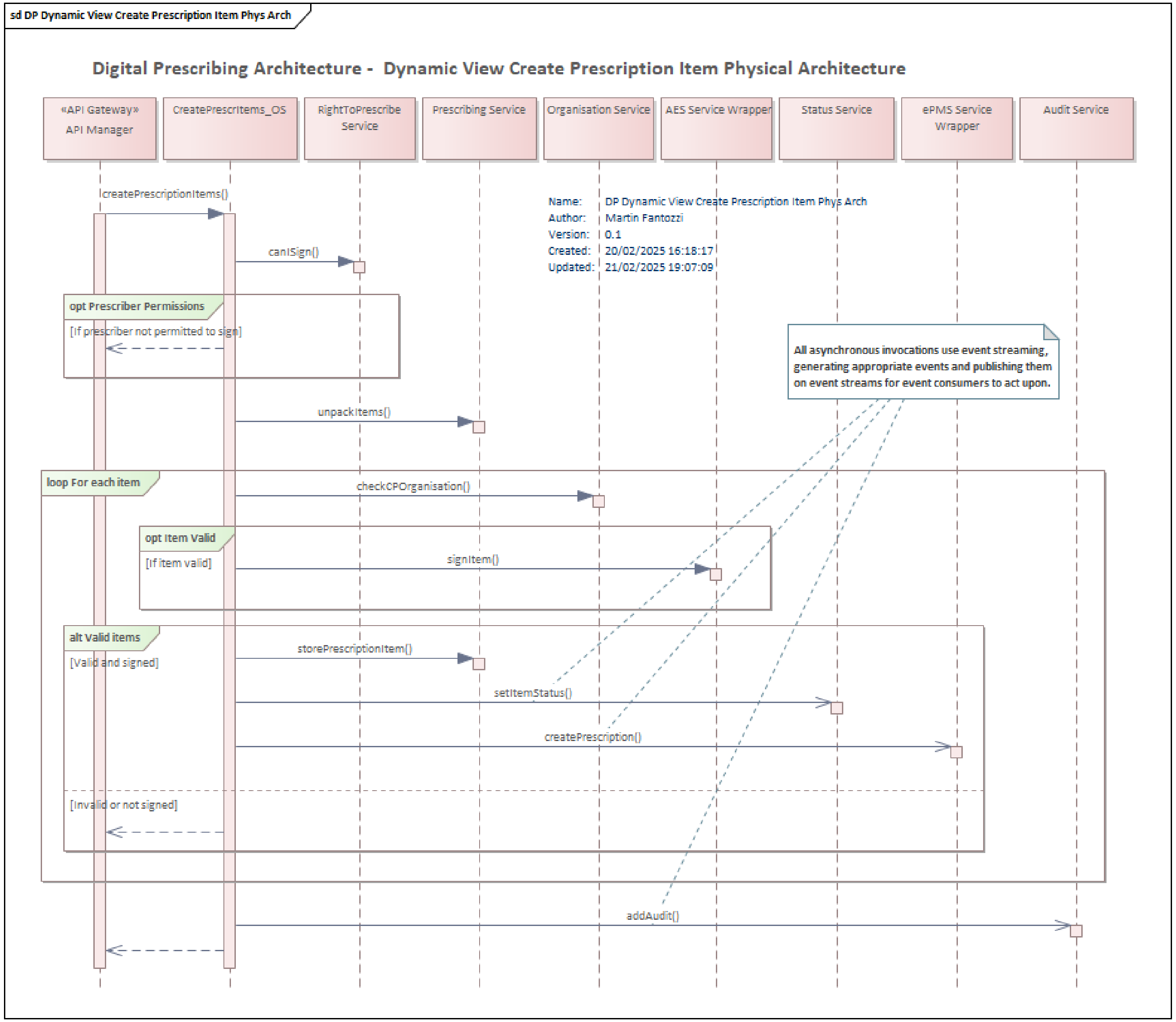
The business process services are the implementation of the business processes defined and elaborated upon by the business analysis team. These services are workflow-based and will invoke the underlying functional services. The workflow pod is shown with two example prescription-related service requests to highlight the dependency of these services on the underlying functional and external services.

The physical architecture of the business process is primarily an implementation of the reference architecture.

Business Process Dynamic Physical Architecture

The following collaborative interaction – the dynamic view of the architecture – shows the service collaboration for the “Create Prescription” workflow service. This shows the key physical components and their collaboration to achieve the aim of the process described.

Create Prescription Collaboration Diagram



Name:	DP Dynamic View Create Prescription Item Phys Arch
Author:	Martin Fantozzi
Version:	0.1
Created:	20/02/2025 16:18:17
Updated:	21/02/2025 19:07:09

All asynchronous invocations use event streaming, generating appropriate events and publishing them on event streams for event consumers to act upon.

Business Function Services Physical Architecture

The business function services are the implementation of the business functions that realise individual process steps and, in some cases, specific low-level capabilities. These services directly implement the individual functions such as validating and storing a prescription.

The physical architecture of the business functions is an implementation of the reference architecture. The deployment pods containing these services are implementations of the pod patterns as shown in the Physical Architecture Realisations section.

Architecture Diagram

Component Descriptions

Prescription Service

This is described above in the Prescription Services Pod Physical Architecture.

Dispensing Service

This is described above in the Prescription Services Pod Physical Architecture.

Status Service

This is described above in the Prescription Services Pod Physical Architecture.

Prescription Search Service

This is described above in the Prescription Services Pod Physical Architecture.

Organisation Service

The organisation service is an implementation of the reference data pod.

Notifications Service

Out of scope for MVP

Patient Personal Preferences Service

Out of scope for MVP

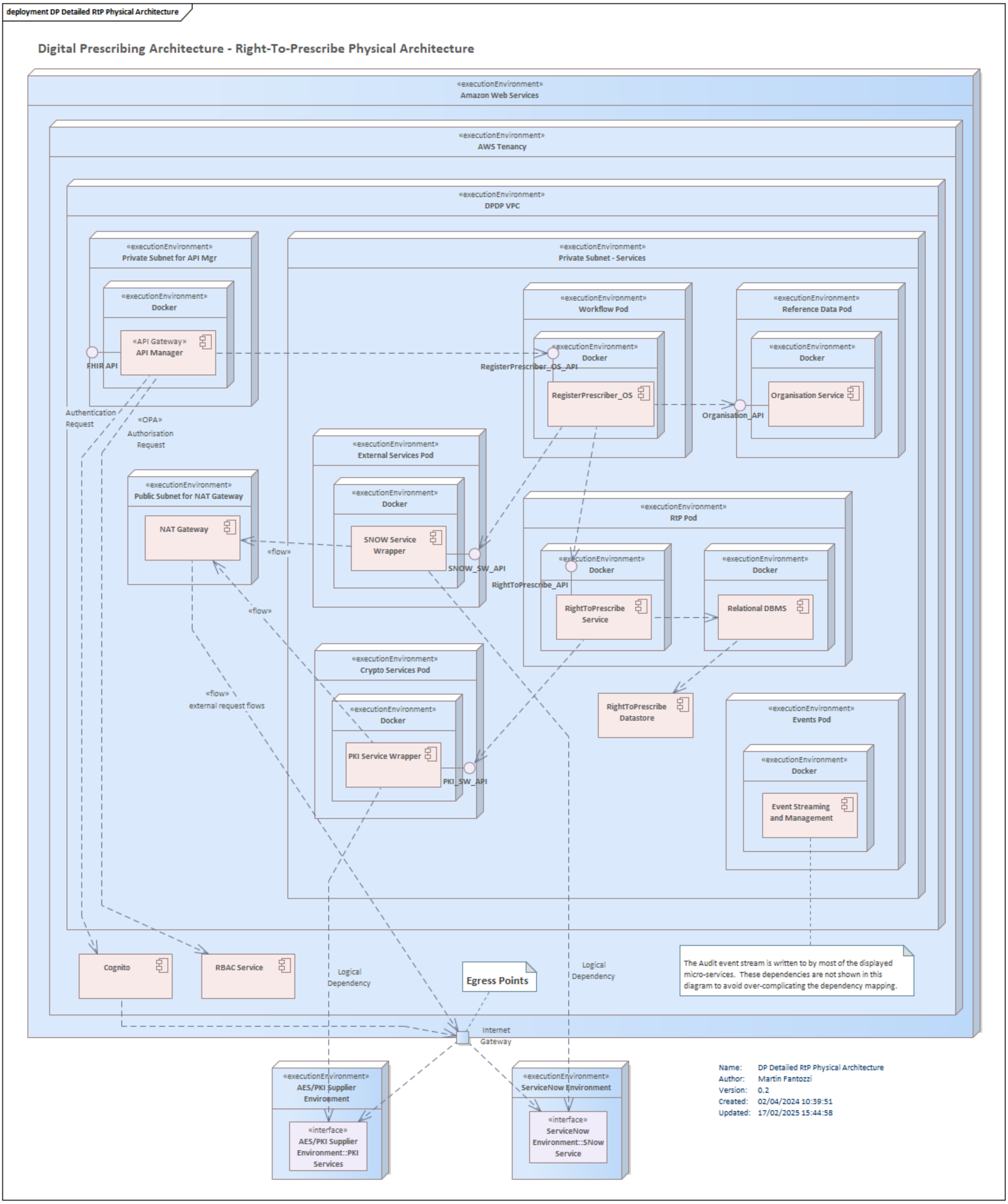
Patient Service

Out of scope for MVP

IAM and the Prescriber Registration Service Physical Architecture

The identity and access management (IAM) platforms and prescriber registration service are intrinsic to the security, signing and non-repudiation capabilities of the solution. These are linked in terms of the business process needed to onboard users, which must cover adding a DPDP user to the core user directory service, ensuring users with the role of prescriber have the appropriate policies within the IAM service to enforce MFA authentication, and adding prescribers to the register of allowed prescribers. The registration of prescribers within DPDP will also trigger the creation of an entry within the AES registration authority (RA) and the generation of x.509 certificates and key pairs. It is expected that setting up a prescriber with the right to prescribe within the registration service will also trigger the creation of a task within the NSS ServiceNow (aka SNow) case-tracking service to request the setup or change of an appropriate prescriber user account within the centralised Entra authentication service.

Architecture Diagram



deployment DP Detailed RtP Physical Architecture
Digital Prescribing Architecture - Right-To-Prescribe Physical Architecture
«executionEnvironment» Amazon Web Services
«executionEnvironment» AWS Tenancy
«executionEnvironment» DPDP VPC
«executionEnvironment» Private Subnet for API Mgr
«executionEnvironment» Docker
«API Gateway» API Manager
FHIR API
«executionEnvironment» Private Subnet - Services
«executionEnvironment» Workflow Pod
«executionEnvironment» Docker
RegisterPrescriber_OS_API
RegisterPrescriber_OS
«executionEnvironment» Reference Data Pod
«executionEnvironment» Docker
Organisation Service
Organisation_API
Authentication Request
«OPA» Authorisation Request
«executionEnvironment» External Services Pod
«executionEnvironment» Docker
SNOW Service Wrapper
SNOW_SW_API
«executionEnvironment» Public Subnet for NAT Gateway
NAT Gateway
«flow»
«flow»
«executionEnvironment» RtP Pod
«executionEnvironment» Docker
RightToPrescribe_API
RightToPrescribe Service
«executionEnvironment» Docker
Relational DBMS
RightToPrescribe Datastore
«executionEnvironment» Crypto Services Pod
«executionEnvironment» Docker
PKI Service Wrapper
PKI_SW_API
«flow» external request flows
«executionEnvironment» Events Pod
«executionEnvironment» Docker
Event Streaming and Management
Cognito
RBAC Service
Logical Dependency
Egress Points
Logical Dependency
Internet Gateway
«executionEnvironment» AES/PKI Supplier Environment
«interface» AES/PKI Supplier Environment::PKI Services
«executionEnvironment» ServiceNow Environment
«interface» ServiceNow Environment::SNow Service

The Audit event stream is written to by most of the displayed micro-services. These dependencies are not shown in this diagram to avoid over-complicating the dependency mapping.

Name:	DP Detailed RtP Physical Architecture
Author:	Martin Fantozzi
Version:	0.2
Created:	02/04/2024 10:39:51
Updated:	17/02/2025 15:44:58

Prescriber Registration Service Physical Architecture

The overall physical architecture of prescriber registration is shown in the diagram above. The physical architecture of the prescriber registration (RtP) service is primarily defined by the reference architecture, however the service is dependent on the AES and PKI services in order to fully register a prescriber and generate appropriate RA entries, key pairs, certificates, etc. In addition it will have a dependency on the organisation service to define which health boards a prescriber is authorised to prescribe on behalf of. It will also use a relational database platform (RDBMS) rather than a data platform optimised for message-style data entities.

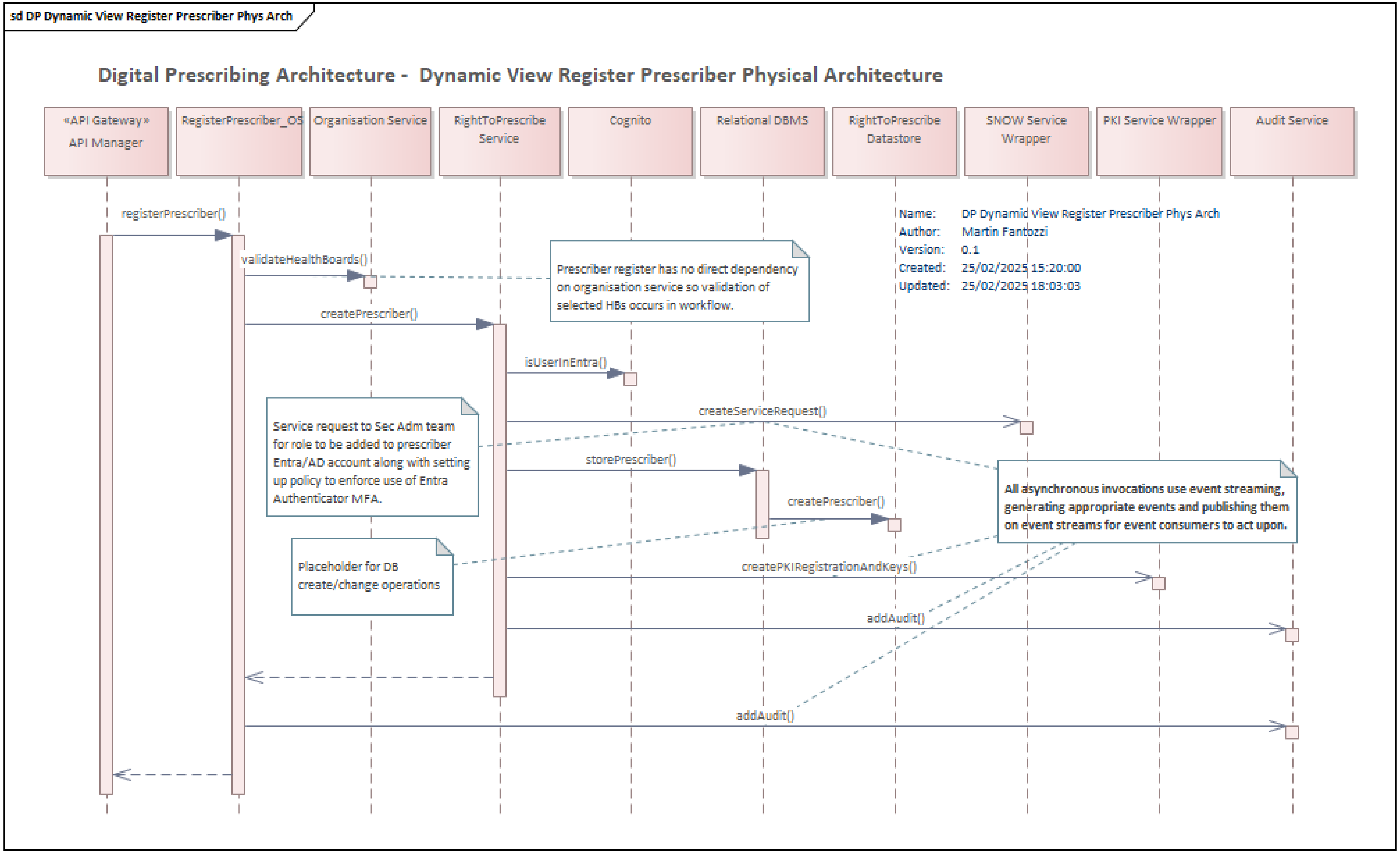
Prescriber accounts in the national user directory service – currently AAD / Entra accessed by AWS Cognito – will require initial setup or further configuration by a business actor in the role of user account administrator. This is an existing role within NSS, and the current intention is that the RtP registration workflow will set up a request within the current NSS ServiceNow instance to perform that task.

Post-MVP it is intended that the RtP service will use provided APIs from professional healthcare workers' registration bodies, such as the GMC, NMC, etc, to verify whether an individual has a current right to prescribe based on their registration details, subject to such a body offering that service. Access to those services will be via façade services and network edge proxies.

Prescriber Registration Dynamic Physical Architecture

The following shows the service collaboration for the "Register Prescriber" internal process service. This shows the key physical components and their collaboration to achieve the aim of the process described.

Register Prescriber Collaboration Diagram



Name: DP Dynamic View Register Prescriber Phys Arch
Author: Martin Fantozzi
Version: 0.1
Created: 25/02/2025 15:20:00
Updated: 25/02/2025 18:03:03

Prescriber register has no direct dependency on organisation service so validation of selected HBs occurs in workflow.

Service request to Sec Adm team for role to be added to prescriber Entra/AD account along with setting up policy to enforce use of Entra Authenticator MFA.

All asynchronous invocations use event streaming, generating appropriate events and publishing them on event streams for event consumers to act upon.

Placeholder for DB create/change operations

Prescriber Registration Architecture Dependencies

The deployment architecture of the RtP service is based on the overall deployment architecture and deployment patterns in [11]. The longer term goal is for the RtP service to be able to access external services provided by professional registration bodies, such as the GMC or NMC. This access will require proxy capabilities at the perimeter of the DPDP domain.

Other Component Descriptions

Authentication and Authorisation

Refer to the Deployment Configuration Considerations for Amazon AWS Cognito and MS Entra section within [11] for more details regarding the integration of the healthcare platform with IAM.

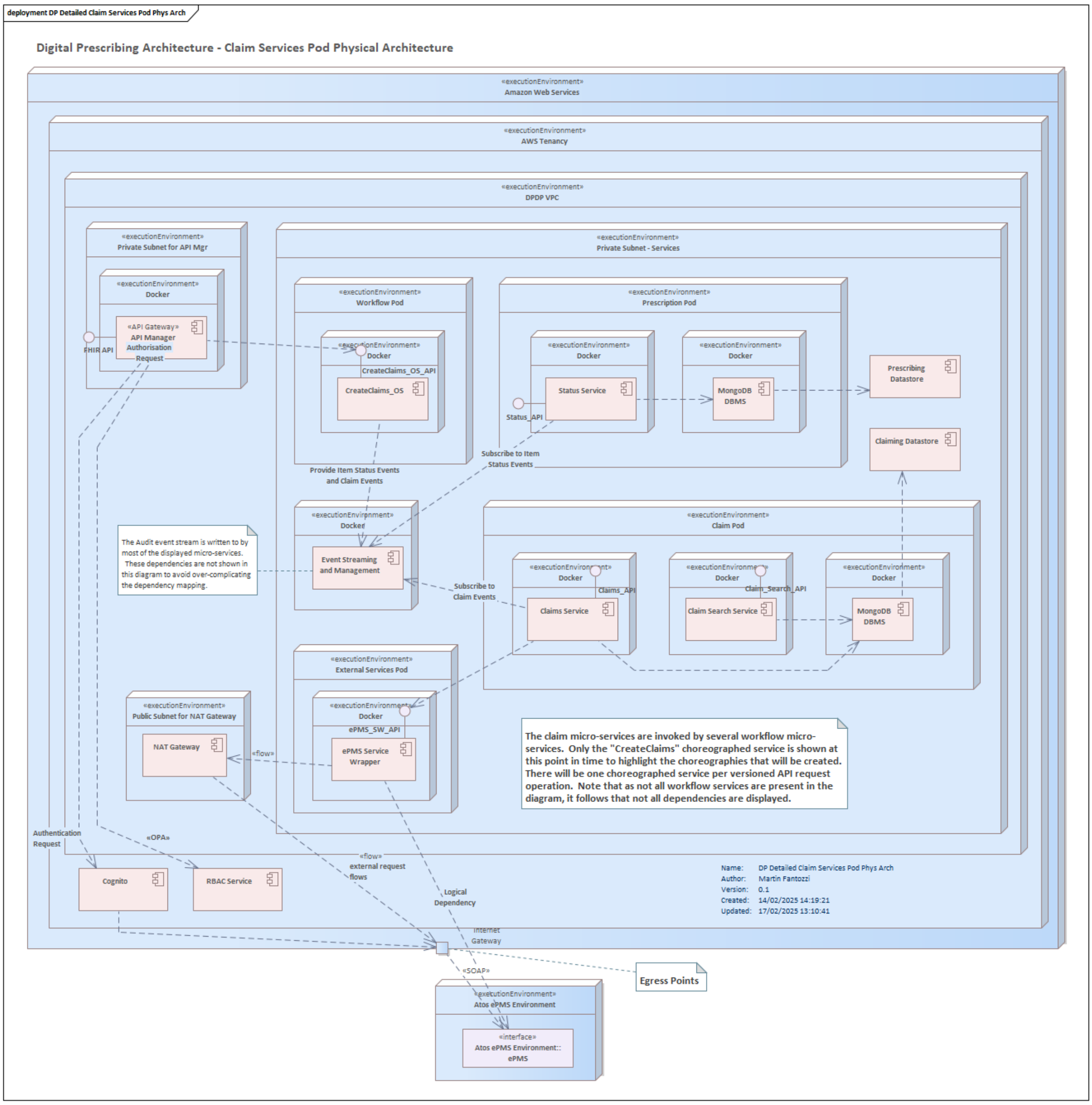
MFA Service – Additional Strong Authentication

Refer to the Deployment Configuration Considerations for Amazon AWS Cognito and MS Entra section within [11] for more details regarding the configuration of MS AAD (aka MS Entra) to enable MFA for the prescriber role.

Claims Service Physical Architecture

The Claim pod services are dependent on a number of other key DPDP services – notably the status service, prescribing data and connectivity to the legacy ePMS service. The following diagram shows not only the internal services deployed within their container context, but also the business process pod and a number of other depended-upon services deployed within their own containers and pods.

Architecture Diagram



The Audit event stream is written to by most of the displayed micro-services. These dependencies are not shown in this diagram to avoid over-complicating the dependency mapping.

The claim micro-services are invoked by several workflow micro-services. Only the "CreateClaims" choreographed service is shown at this point in time to highlight the choreographies that will be created. There will be one choreographed service per versioned API request operation. Note that as not all workflow services are present in the diagram, it follows that not all dependencies are displayed.

Name:	DP Detailed Claim Services Pod Phys Arch
Author:	Martin Fantozzi
Version:	0.1
Created:	14/02/2025 14:19:21
Updated:	17/02/2025 13:10:41

Claim Pod Physical Architecture

Component Descriptions

Claims Service

The claims service implements the read, creation and update capabilities for claim items.

Claims Search

The claim search service provides claimed item search capabilities using different combinations of search criteria. All search results will be returned as a full set of data for the item, including medication and claim information, or partial sets of data where the search has been invoked by a business actor in a role that does not allow direct access to confidential patient information.

MongoDB DBMS and Claiming Datastore

The claims services use the MongoDB data persistence platform. This will be deployed in its own Docker container within the Claim pod. The datastore will exist outside of the managed containers.

AES and PKI Services Physical Architecture

The AES service is fundamental to the paperless prescribing process and uses a set of public-key infrastructure (PKI) components to provide a certificate-based encryption service with non-repudiable digital signatures for each prescribed item. However the PKI and AES signing services will be sourced externally and their internal architecture will not be further elaborated upon within this section.

These services will be accessed via a wrapper service that realises the external service pod pattern – refer to the Reference Physical Architecture – Internal Pod and Service Architecture and Physical Architecture Realisations sections.

Architecture Diagram

Component Descriptions

PKI Components

Registration Authority (RA)

Certificate Authority (CA)

Validation Authority (VA)

AES Services and Components

Signing and Verification Service

Signing API

Signature Verification API

Client Systems Physical Architecture

The DPDP service is based on an API-centric architecture, with very few user interfaces of its own. The majority of interactions with the service occur via a FHIR API and are initiated within external client systems. These external systems are deployed by their respective vendors for their own customers, within external domains, and with no requirement for deployment to a cloud tenancy or data centre for which the DPDP operational team will have access or responsibility. This means that there are few DPDP physical client artefacts to be created and deployed within the overall programme domain.

The internal clients' physical and deployment architecture is described in [11].

Architecture Diagram

Component Descriptions

Internal NHS Client Architecture

Need UI reference piece here

External Client Architecture

GP IT

PMR

Patient-Facing Application Service

Refer to the Patient-Facing Application subsection for information.

Direct Messaging

Refer to the Direct Messaging Service subsection for information.

Patient-Facing Application – Physical Architecture

The patient-facing application domain contains the services, components and platforms necessary for the general public to have access to DPDP services and additional externally-sourced information. This access will be, at a minimum, via a mobile app, however this could be extended to include a website suitable for PCs and desktops at some point.

Access to the app will require a member of the public to download the app to a compatible mobile device, register to use the app by verifying their identity and setting up a user account, then sign into the app. It is likely that this will be an extension of the Scots govt Digital Front Door initiative.

Architecture Diagram

Component Descriptions

Mobile Device App

Website for Prescription Information

Verification Service

Authentication Service

Separated Public API

The app will use a separate, public API tailored to the needs of the healthcare app. This API will access the DPDP services via the same healthcare platform as the other DPDP APIs and clients.

Direct Messaging Service – Physical Architecture

The requirement for direct messaging is that it should allow direct, “chat” style messaging from one user of DPDP to another. This is a post-MVP requirement that will be dependent on the deployment requirements for a selected product.

Component Descriptions

Direct Messaging Service

Direct Messaging API

Direct Messaging User Interface

Link to Prescription Items

Access to DPDP information is expected to be via a published DPDP API.

General Services and Components – Physical Architecture

This section contains the “side-bar” components that will be used by all new services and components within the overall DPDP service. Those components are principally focused on common functions such as service health, security event monitoring and alerting, appropriate usage monitoring. They are also common services used across the broader technology landscape and are not restricted to use within DPDP.

The physical architecture of the general services and components is defined by the reference architecture.

Architecture Diagram

Component Descriptions

Service Monitoring

Security Information and Event Management (SIEM)

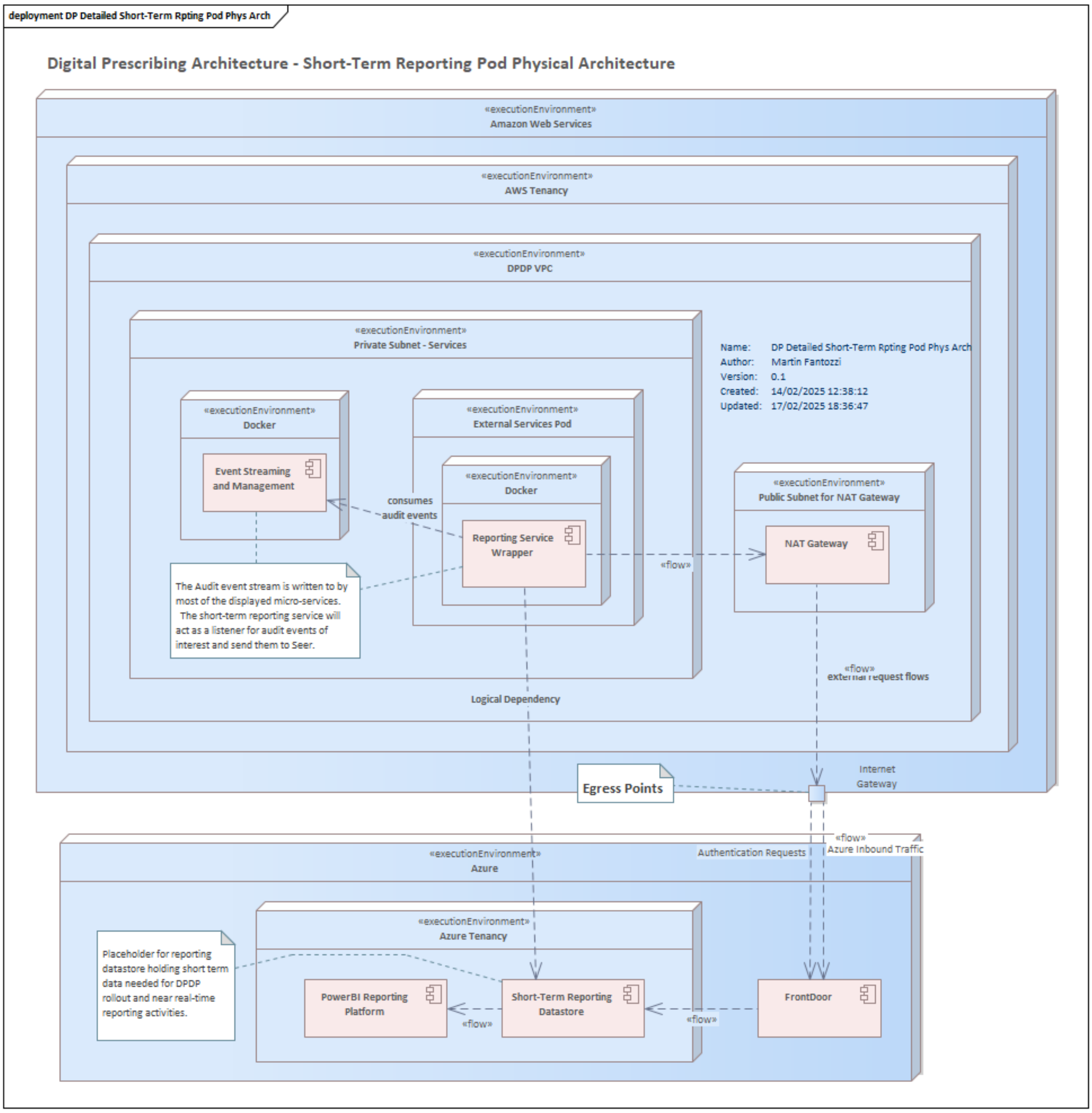
Appropriate Use Monitoring

Service Request

Short-Term Reporting Physical Architecture

The Prescription Services pod and dependencies highlight the physical architecture of most of the key functional areas within the overall DPDP service. The following diagram shows not only the internal services deployed within their container context, but also the business process pod with appropriate example workflow services and a number of other depended-upon services deployed within their own containers and pods.

Architecture Diagram


deployment DP Detailed Short-Term Rpting Pod Phys Arch
Digital Prescribing Architecture - Short-Term Reporting Pod Physical Architecture
«executionEnvironment» Amazon Web Services
«executionEnvironment» AWS Tenancy
«executionEnvironment» DPDP VPC
«executionEnvironment» Private Subnet - Services
«executionEnvironment» Docker
Event Streaming and Management
«executionEnvironment» External Services Pod
«executionEnvironment» Docker
Reporting Service Wrapper
consumes audit events
«flow»
«executionEnvironment» Public Subnet for NAT Gateway
NAT Gateway
«flow» external request flows
Logical Dependency
Egress Points
Internet Gateway
Authentication Requests
«flow» Azure Inbound Traffic
«executionEnvironment» Azure
«executionEnvironment» Azure Tenancy
PowerBI Reporting Platform
«flow»
Short-Term Reporting Datastore
«flow»
FrontDoor

Name: DP Detailed Short-Term Rpting Pod Phys Arch
Author: Martin Fantozzi
Version: 0.1
Created: 14/02/2025 12:38:12
Updated: 17/02/2025 18:36:47

The Audit event stream is written to by most of the displayed micro-services. The short-term reporting service will act as a listener for audit events of interest and send them to Seer.

Placeholder for reporting datastore holding short term data needed for DPDP rollout and near real-time reporting activities.

External Services Pod - Reporting Service Physical Architecture

Component Descriptions

Reporting Services Wrapper

The Reporting Services Wrapper is a façade service that will consume item status events, and possibly audit events for creation and update of prescribers and organisations. These will be sent to a short-term aggregate reporting store hosted within Azure and part of the Seer platform. It has not been decided as yet as to whether frequent small batches of transactions or pre-aggregated data will be sent to Seer.

Azure Tenancy – NSS Seer Deployment

Short-Term Reporting Datastore Search

This is a short-term aggregate reporting store hosted within Azure and part of the Seer platform. It will host relatively short-lived and fine-grained aggregate information relating to the rollout and uptake of DPDP digital prescribing.

PowerBI Reporting Platform

The PowerBI platform will be used to extract and present data from the short-term DPDP data for use by internal NHS stakeholder groups.

Existing NHS Systems

This subsection contains the specific NHS Scotland systems and services that currently are fed by ePMS file transfers and which will be replaced by direct DPDP data integration. This data integration will use common, flexible and resilient ETL platform technology. However for the initial MVP release DPDP will continue to use ePMS as the interface point.

Refer to the Data Architecture of existing NHS systems in [10] for details of the data integration platform and physical data architecture.

Architecture Diagram

Placeholder – to show use of ETL within AWS/DPDP context.

Component Descriptions

nDCVP

The nDCVP data tier will be populated using the ETL toolset. There has been no decision as to whether this will be a real-time update or how often this update will run otherwise.

Prescription Information Service (PIS) and PRISMS

The PIS datastore will be populated using the ETL toolset. This update will run on a schedule to be defined by the SEER programme.

Dynamic Physical Reference Architecture

Synchronous Invocation– Service Collaboration and Dynamic Behaviour

Orchestration Service Collaboration

Function Service

External Service Collaboration