Medical Care Collections Fund (MCCF) ePharmacy

Transactions Applications Suite (TAS)

US4109 ePharmacy Realtime Reporting Electronic Claims Management Engine (ECME)

System Design Document

Logo for the Department of Veterans Affairs, Office of Information and Technology, Product Development, including the official seal of the Department of Veterans Affairs


Department of Veterans Affairs

May 2018

Version 0.0.2

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
| 05/17/2018 | 0.0.1 | Initial creation of document | Tony L. Howard - Halfaker and Associates |
| 05/21/2018 | 0.0.1 | Convert to US4109 | Keith Oulson – Halfaker and Associates |
| 05/31/2018 | 0.0.2 | Prepare for further development | Keith Oulson – Halfaker & Associates |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1 Introduction 4

1.1 Purpose 4

1.2 Scope 4

2 Architecture 5

2.1 Enterprise Context 5

2.2 Architecture Overview 6

2.3 Rules Engine 8

3 Design 10

3.1 Assumptions, Constraints, Dependencies and Risks 10

3.2 Business Process 10

3.3 Application 10

4 Database 14

4.1 Vista 14

4.2 MCCF Datastore 14

5 Technical 15

5.1 System 15

5.2 Network 15

5.3 Security 16

5.4 Administration 16

Table of Figures

[Figure 1 - TAS Platform Components 5](#_Toc515523510)

[Figure 2- Architecture Overview Diagram 6](#_Toc515523511)

[Figure 3 - Rules Engine Architecture 8](#_Toc515523512)

[Figure 4 - Use Cases 10](#_Toc515523513)

[Figure 5 – Real Time Data Retrieval 11](#_Toc515523514)

[Figure 6 - Tas Core Environment Overview 15](#_Toc515523515)

[Figure 7- Tas Core Mag Dev/Test Network Overview 15](#_Toc515523516)

[Figure 8 - IAM Service Security Overview 16](#_Toc515523517)

# Introduction

This System Design Document (SDD) describes the System Design for the ePharmacy Reporting design of an MCCF TAS platform business service in support of the selection and processing of Recent Transactions Report in the Electronic Claims Management Engine (ECME) ePharmacy Reporting Suite.

## Purpose

The purpose of this (SDD) is to define the Architecture, Design, Business Processing, Customer Application Development, Database and other Technical needs that will be used to create business services in support of the selection and processing of ECME Reports.

## Scope

This SDD describes the detailed design and use of the MCCF TAS platform components in support of the ECME ePharmacy reports user story US4109. This document is intended for all parties requiring such information, including business stakeholders, software developers, system designers, testers and anyone else responsible for implementing this design.

# Architecture

This diagram shows the planned TAS components and their relationship to each other. The design in the following slides and highlighted components are specific to the Business Rules Engine release of TAS. Architecture and Design for the use of other TAS components are documented elsewhere.

## Enterprise Context

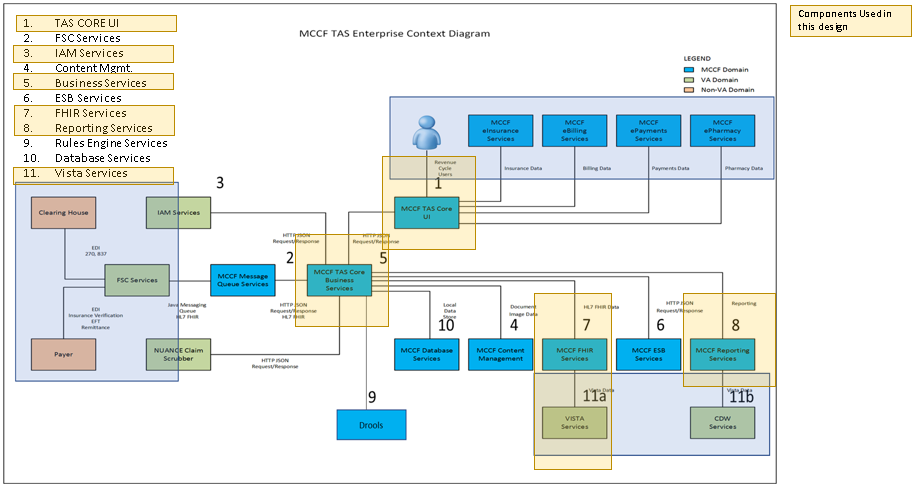


Figure 1 - TAS Platform Components

## Architecture Overview

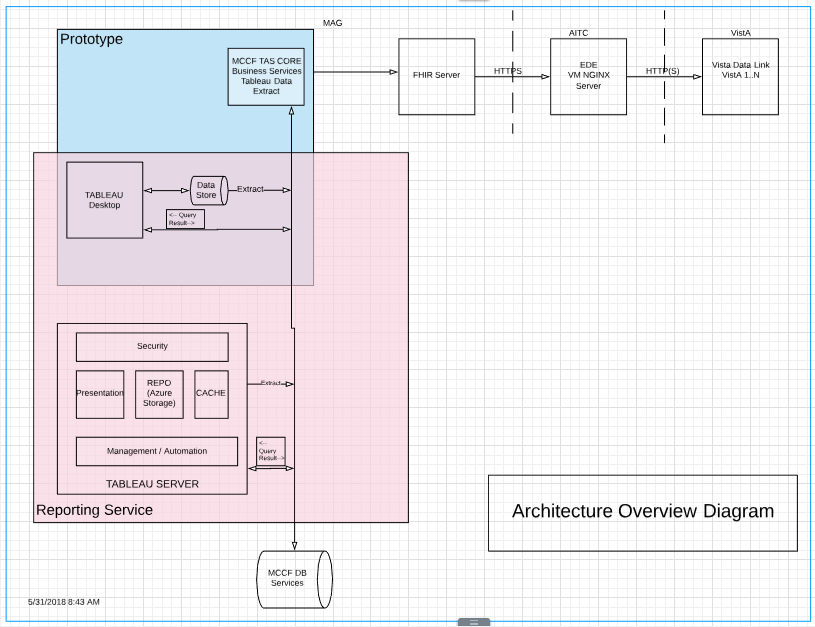


Figure 2- Architecture Overview Diagram

### WEB User Interface

The WEB User Interface component of the MCCF TAS platform employs a web application architecture that includes an Angular Single Page Application (SPA).

### Financial Service Center (FSC) Interface Shared Service

The TAS Platform FSC I/F component provides a REST Based interface using and internal that will send and receive FHIR messages between TAS and the FSC

### VA Identity and Access Management (IAM) Shared Service

The Department of Veterans Affairs (VA) has implemented an Identity and Access Management (IAM) Service that provides access to VA information, resources, and services to improve timeliness and promote ease of access for all VA users. The TAS Platform has integrated with the IAM using it as the main service to provide for Single Sign on Authentication.

### Content Management Shared Service

The content management component is a set of related services that are used to provide and manage digital content. The system will facilitate the management and incorporation of documents and other digital assets into the overall architecture of the TAS Platform.

### TAS Core API Shared Service

The TAS Core API is a shared component of the TAS Platform that allows for the creation of sharing java script based services. The Core API provides for business service development and connection to shared resources by way of providers that tie API’s to underlying data structures. This Core API is the basis for any combination of services that need to be provided for within the TAS Platform. It is a baseline component that can be reused and made available for development of business services.

### Mule ESB Shared Service

Mule ESB is an internal runtime component of the TAS platform that allows for the orchestration of TAS services. The Anypoint studio is the component that provides a GUI based interface that allows developers to create workflows and other constructs that tie services together. Once the orchestrations are completed within the TAS Platform they are uploaded to the Mule ESB for either continuous or timed execution.

### FHIR Server Shared Service

The FHIR server component provides the TAS platform with a consistent, compliance driven REST interface. The server is JAVA based and resides within the boundary of the TAS Platform and provides the main means of access to the Vista data sources. JavaScript Object Notation (JSON) is the main data format used internally within the TAS Architecture and provides a means of moving objects between services. There is a subset of FHIR resources that have identified and are being used by the TAS Platform.

### Reporting Server Shared Service

The reporting service within the TAS Platform will provide both reporting and visualization of data and is based on Tableau Server. The Shared Service is based on the Tableau Server component and provides for access and delivery of reports and visualizations to the end user via an embedded JavaScript that links the Web UI to the Tableau Server.

### Rules Engine (Drools) Server Shared Service

The Rules Engine server is based on an Open Source version of the popular Drools server that provides for the receipt of facts, comparison and rules set application and return of responses. The server is named Knowledge is everything or (KIE) and provides access via HTTP JSON payload based service calls to the actual rules set definitions and responses. The services are created using a GUI based modeling and development environment. (See 2.2 Rules Engine for more information)

### MCCF Datastore Shared Service

The MCCF Datastore is a no-sql data store that provides for the storage of a key and value (entire JSON document structure). The current data store uses defined tables and is a robust storage capability called Microsoft Azure Tables and is available from directly from Microsoft Azure Cloud service provider.

### Vista Instance Shared Service

The Vista Instance shared service is based on an embedded Mumps HTTP Listener and provides delivery of data from standard Fileman structures and/or RPC’s and uses a RESTFULL like interface that provides CRUD capabilities and Security control integration in concert with the VA IAM Shared Services.

## Rules Engine

The Rules Engine is made up of two parts, the rules modeling and development environment which can be shared or used standalone by a developer and the Knowledge is Everything (KIE) Server. The main part used during the creation and development of a rules engine component will be the Drools Workbench which provides a GUI based modeling/development environment. After creation of the component it is then packaged and deployed to a KIE Server where interaction is limited to HTTP calls with Java Server Object Notation Objects embedded in the delivery. These delivered objects will represent a Service that contain facts that will read in an HTTP JSON request message, evaluate based on the rules defined and then return a HTTP JSON response message will be generated and returned. The diagram below shows the architecture of the Rules Engine and how it will interact with the operational environment.

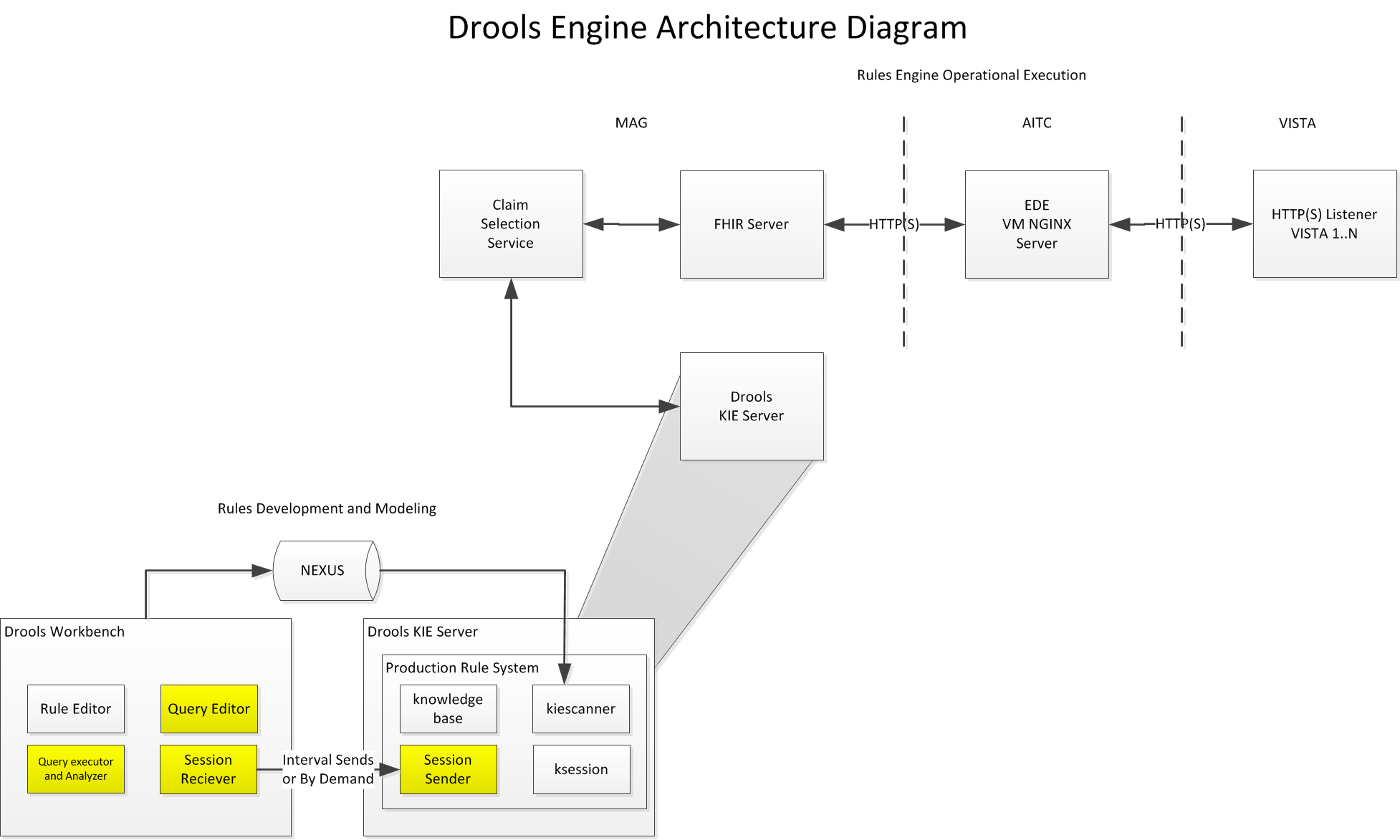


Figure 3 - Rules Engine Architecture

### Claims Selection Service

This service will be a TAS API that will select transactions, create and provide facts for processing by the Rules Engine Component.

### Claims Processing Service

This service will also be a TAS API that will select transactions, create and provide the 837 Single Claim Multi Resource Bundle to be sent to the TAS-FSC for transmission to the FSC for additional processing

### Rules Development and Modeling

During the design of rules the process of modeling the facts, creating the rules and testing is completed separate from the distribution of the Rule Engine Component

### Drools KIE Server

The Knowledge is Everything (KIE) server is where the developed rules engine components will be placed in order to allow access send facts and return answers.

### FHIR Server

Resource server attached and providing data from the multiple vista instances.

### EDE Gateway from MAG

This will be a HTTP(S) instance of NGINX that will allow for traffic from the MAG to flow through AITC and the vista instances.

### Vista Data Instances

These are the 130 instances of VistA that will be accessed to support data retrieval

# Design

## Assumptions, Constraints, Dependencies and Risks

* A - An initial investigation and related findings are documented
* C - Must use a TRM approved product for use in the TAS platform
* D – AoA for the use of Drools as the Business Rules Engine – Completed
* R – If Fields within the individual Vista databases are non-existent, dissimilar and or do not have the same context additional work may need to be completed in order to provide the correct mapping of information.

## Business Process

The typical Use case defines a list of actions or event steps that show the interactions between a role or actor and a system that are used to achieve a goal. In this instance, users will be looking at real time information returned to their request.

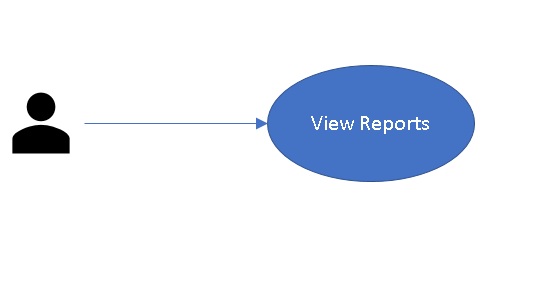


Figure 4 - Use Cases

## Application

### Recent Transaction Selection Service

The ePharmacy Development Team may design and develop a TAS Core business service that could facilitate retrieval via an HTTP request to the FHIR Server that would in turn make a request to the M Web Server listener (VistA Data Link VDL) to return requested ECME information. The results of this request to VistA could be a list of ECME entries (IENs of the xxxxxx file) that need to be included in the Recent Transactions ECME reports.

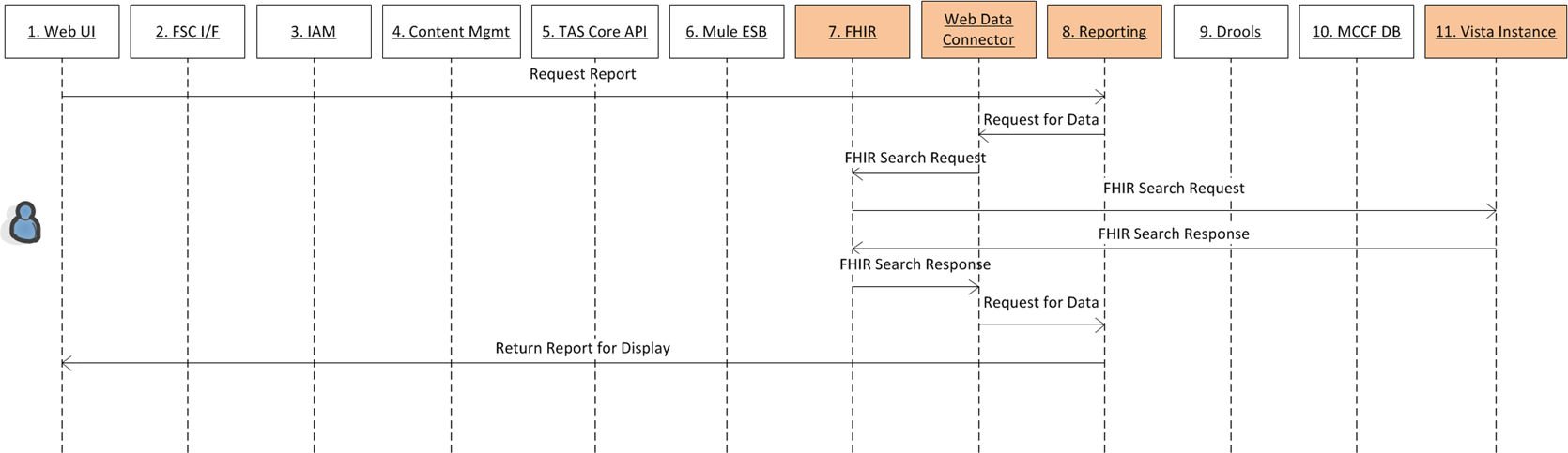


Figure 5 – Real Time Data Retrieval

* 1. Start Processing
* 2. Request Report
* 3. Request for Data in the Report
* 4. FHIR Search Request
* 5. FHIR Search Request for Needed Data in VistA instance
* 6. FHIR Search Response of Needed Data from VistA instance
* 7. FHIR Search Response
* 8. Requsted Data Back to Reporting Service
* 9. Return Report for Display

#### Test Definitions

* Test that all components are available and ready for processing. Each component should have a health check available to execute and return the current status.
* Multiple CURL Recent Transaction Selection Service JSON Message Posts ready to be submitted and executed through the system to exercise the Primary Components used in the detailed design.

#### Message Structures

##### Payload Type

Format (JSON), Structure, Elements, Attributes

* A Curl Post based JSON Message to start the Initial Action.
* A JSON Based resource for the Recent Transactions Report Data

##### Define the Input

What are the pre-conditions of the operation/service

* The initiation of the criteria selection would be initialized by a cron job calling the HTTP FHIR Server search action via a CURL Call from the command line. This call to the HTTP FHIR Server would query the FHIR server selecting recent transaction information from Vista and returning a list of IEN’s, the list would then be passed back from the VISTA HTTP Listener (VDL) though the FHIR server.
* During processing, the list of IEN’s would be accessed during a loop that would initiate an additional HTTP Request of the FHIR Server to retrieve, construct and post a Recent Transaction Multi Resource (or other suitable method) bundle to the TAS FSC component for further processing.

##### Define configuration elements

Use flexible lookup patterns, Do not hardcode anything, Use a database table or a central configuration repository

* FHIR Resources
* VistA Datastore
* TAS FSC Component

##### Define the Output

What are the post-conditions of the operation

* The initial list that would be returned would include a list of IEN’s for processing.
* The Claim structured as a Single Claim Multi Resource Bundle extracted and sent to the TAS FSC Component.
* Response of Successful Transmission or need for Retransmission would be provided.

##### Define the transaction

Pseudo code the Solution

#### Define the Resources

##### Names and Descriptions

Definition and Setup; Interface Definitions; API’s, Libraries, Database Components; File Systems, Logs

##### Define the resources used during Transaction

Log output (be consistent), Error codes (be consistent), Interfaces, Database Tables

#### Define the Policies

##### Types of Policy

Security, Functional, Regulatory, Performance, Constraint Definition

##### Define policies and constraints

Regulatory, Business, Architecture

#### Test Definitions

* Test that all components are available and ready for processing. Each component should have a health check available to execute and return the current status.
* Multiple CURL Claim Selection Service JSON Message Posts ready to be submitted and executed through the system to exercise the Primary Components used in the detailed design.

#### Message Structures

##### Payload Type

Format (JSON), Structure, Elements, Attributes

* A Curl Post based JSON Message to start the Initial Action.
* A JSON Based resource for the ePharmacy requested Data

##### Define the Input

What are the pre-conditions of the operation/service

* The initiation of the criteria selection would be initialized by a cron job calling the HTTP FHIR Server search action via a CURL Call from the command line. This call to the HTTP FHIR Server would query the FHIR server selecting ePharmacy Vista and returning a list of IEN’s, the list would then be passed back from the VISTA HTTP Listener (or VDL) though the FHIR server.
* During processing, the list of IEN’s would be accessed during a loop that would initiate an additional HTTP Request of the FHIR Server to retrieve, construct and post a Recent Transaction Multi Resource bundle (or other suitable method) to the TAS FSC component for further processing.

##### Define configuration elements

Use flexible lookup patterns, Do not hardcode anything, Use a database table or a central configuration repository

* FHIR Resources
* VistA Datastore
* TAS FSC Component

##### Define the Output

What are the post-conditions of the operation

* The initial list that would be returned would include a list of IEN’s for processing.
* The Recent Transactions structured as a Multi Resource bundle (or other suitable method) extracted and sent to the TAS FSC Component.
* Response of Successful Transmission or need for Retransmission would be provided.

##### Define the transaction

Pseudo code the Solution

1. The TAS Core business service may periodically send an HTTP search request to the FHIR Server which in turn will make a request of VISTA and check for TBD….
2. Based on the search request previously executed the list of IEN’s could be used to execute the process to TBD….
3. It is still to be determined (via the design process) if there are several HTTP requests back and forth to create the xxx FHIR Transactions, or if one request for required xxx data is sufficient for each xxx FHIR Transaction creation.

#### Define the Resources

##### Names and Descriptions

Definition and Setup; Interface Definitions; API’s, Libraries, Database Components; File Systems, Logs

##### Define the resources used during Transaction Processing

Log output (be consistent), Error codes (be consistent), Interfaces, Database Tables

#### Define the Policies

##### Types of Policy

Security, Functional, Regulatory, Performance, Constraint Definition

##### Define policies and constraints

Regulatory, Business, Architecture

# Database

## Vista

Schema, Object Combination, Duplication of Data, Joins, Optimize, Complex Aggregations

## MCCF Datastore

The Datastore might be used as a SYNC DB to replicate VistA data in the ePharmacy processing of Recent Transactions in order to achieve better performance for real time functionality.

# Technical

System, Network, Security & Administration

## System

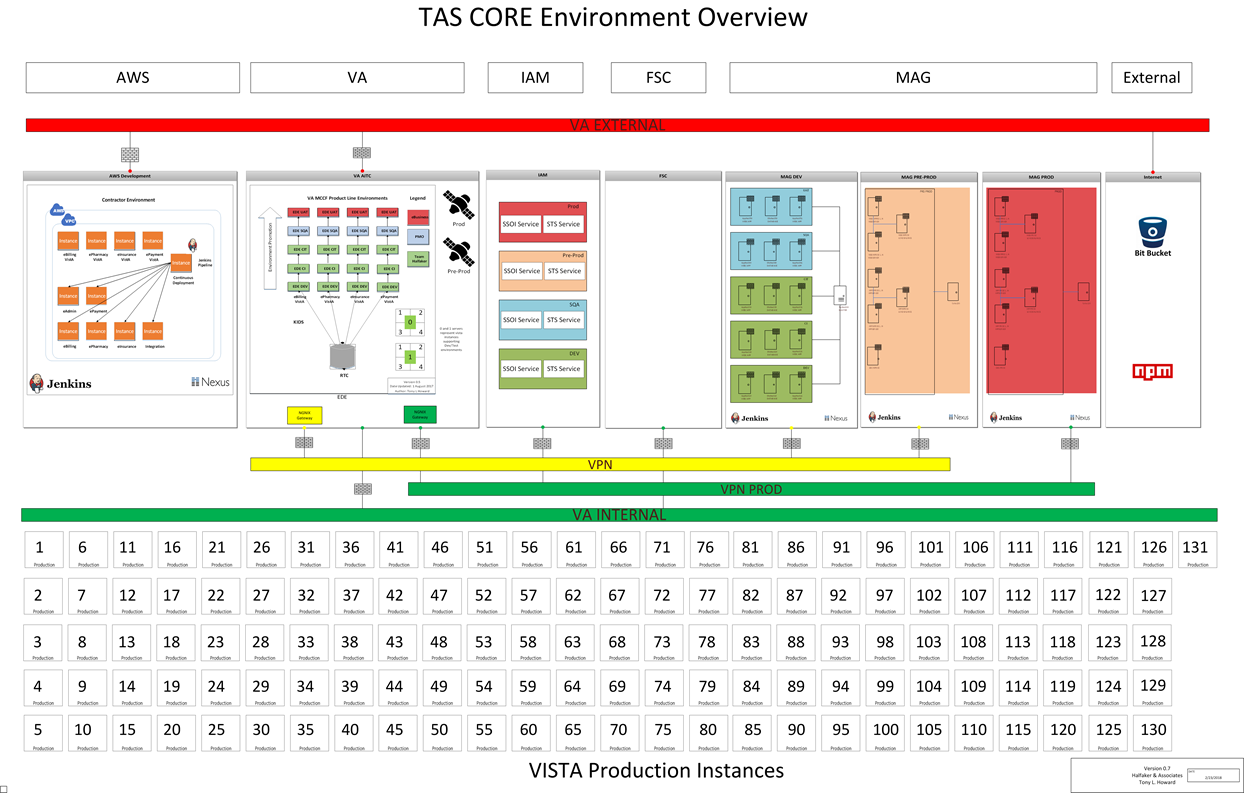


Figure 6 - Tas Core Environment Overview

## Network

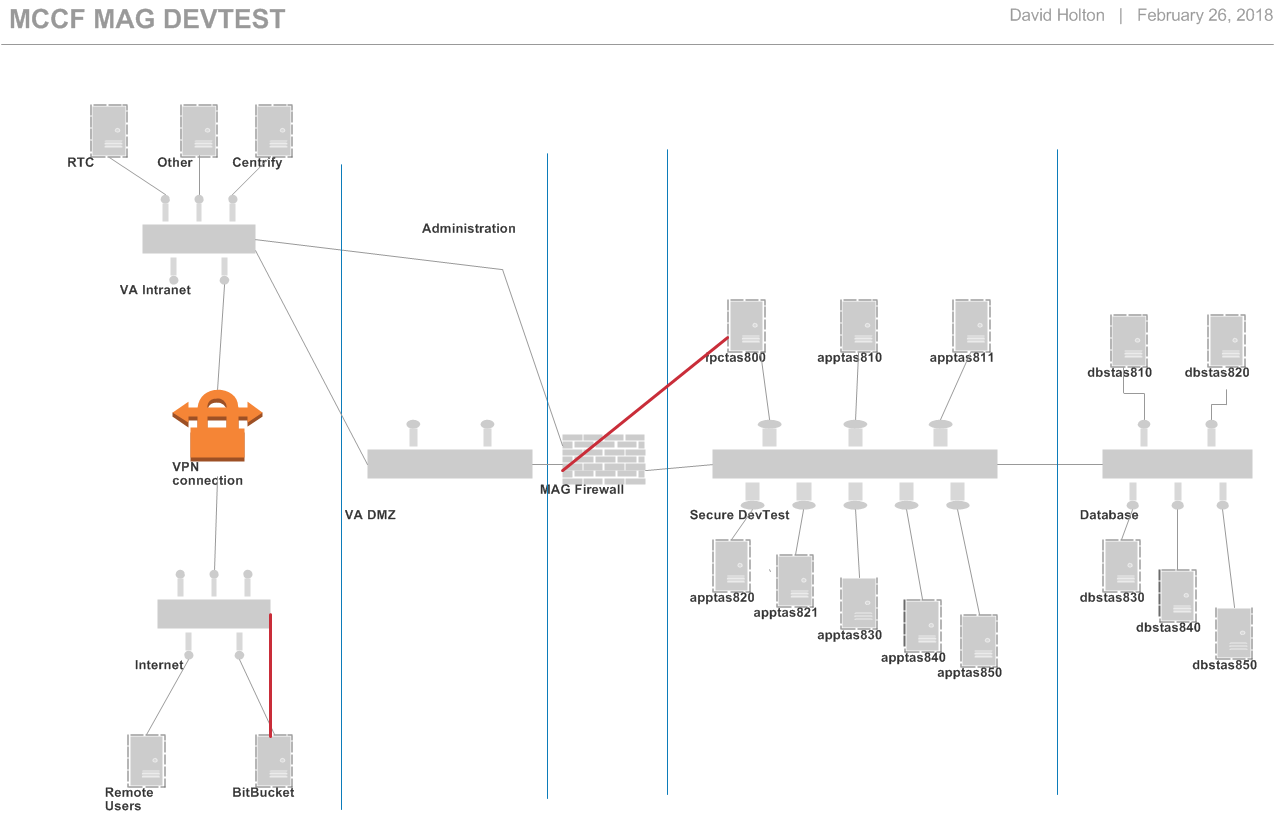


Figure 7- Tas Core Mag Dev/Test Network Overview

## Security

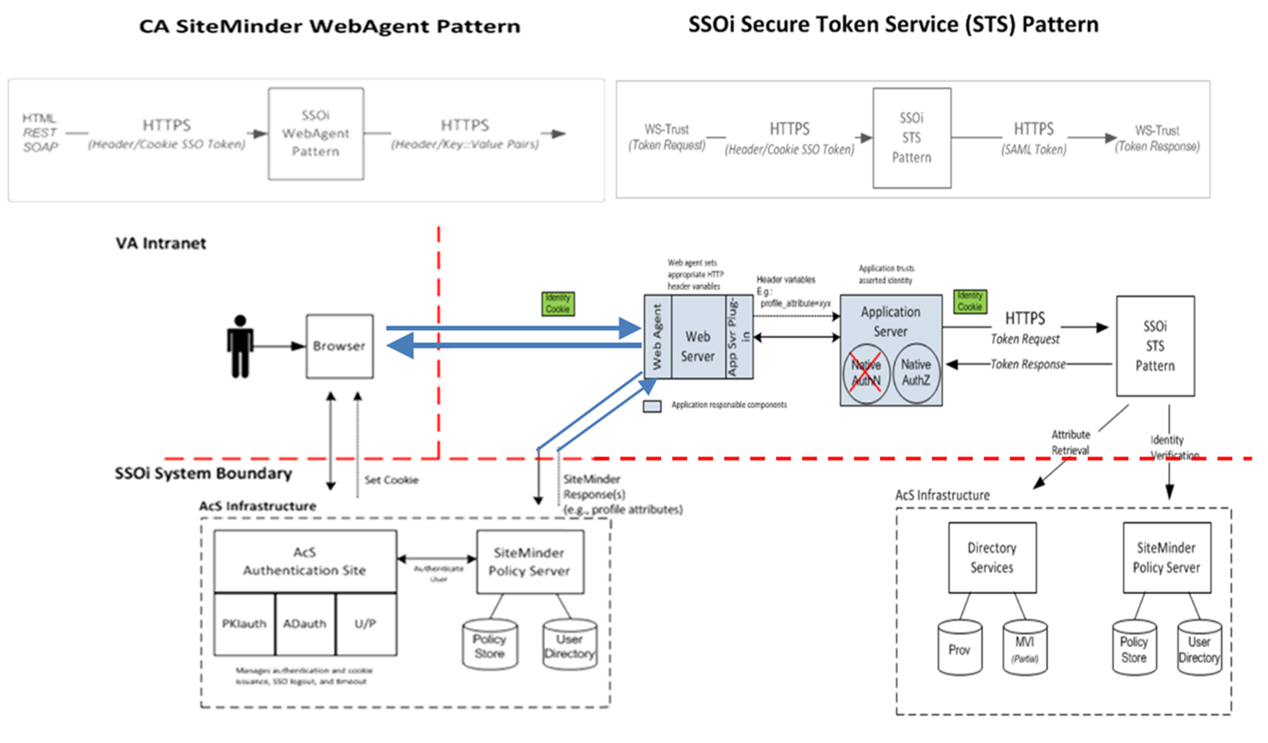


Figure 8 - IAM Service Security Overview

## Administration

We understand the need for, and have CM and Administration support of the current development systems and overall maintenance. A longer term strategy for the administrative sustainment of the business rules engine still needs to be determined.