CAN 5.5

Requirements Document

APRIL 29, 2021

AVANSEUS TECHNOLOGIES PVT. LTD.

Table of Contents

Revision History

Date	Created / Modified by	Reviewed by	Comments
29-04-2021	Naveen / Sandeep Singh	Chiranjib Bhandary	Draft

Focus

The main focus of this release is to enable CAN application to run on cloud native Kubernetes platforms. Cloud native Kubernetes platforms enable hardware usage optimizations, simplify deployment, scale the application on need basis, ensures high availability and most importantly, allows micro-services based architecture. The release also includes remaking the UI using React framework for better responsiveness and major features like policy based RoE, ServiceNow integration and Kafka integration along with other additional features. Full list of features we have productized in this release are produced below.

Cloud Native Features:

1. Kubernetes Based Deployment
2. Elasticstack Integration in Openshift
3. Pod Logging
4. Horizontal Pod Auto Scaling
5. Helm Charts for Easy Installation
6. Integration with Istio Service Mesh
7. Integration with Monitoring softwares (Kiali, Prometheus and Grafana)

3rd Party Product Integrations:

8. Service Now Integration
9. Kafka Integration
10. Memcached Tool Integration for Centralized Cache
11. NFS Server Data Storage for Pods
12. Mongo DB TLS Integration

Other Features:

13. Advanced RoE and Ticket Matching
14. Web Security Configuration Porting from Apache to Nginx
15. Tomcat Clustering for Session Management in CAN & CAS
16. UI Porting with React Framework
17. Mongo DB Sharding
18. Mongo DB Version Upgrade from v3.4.6 to v4.4.5
19. Single login session for a user
20. Workorder integration in Parser
21. Parser screen enhancements
22. Java Security Manager Enhancement
23. Re-scheduling of Trigger based on UI Cron Pattern Configurations
24. Prediction as a service
25. REST API for prediction delivery

Terminologies

Requirements are classified based on type & priority.

A. Requirement Types

Requirement Type	Definition
Business	Business requirement deals mainly with business goals and stakeholder expectations and tells us about the future state of the product and why the objective is worthwhile.
Functional	Functional requirements are much more specific and detailed compared to business requirements. They outline how a product will support business requirements and specify the steps on how the requirement will be delivered.
Non-functional	The non-functional requirement elaborates a performance characteristic of the system. These requirements fall in areas such as accessibility, documentation, efficiency, disaster recovery, security etc.,

B. Requirement Priorities

Priority	Semantics
Critical	A critical requirement without which the product is not acceptable to the stakeholders
Important	A necessary but deferrable requirement which makes the product less usable but still functional
Desirable	A nice feature to have if there are resources but the product functions well without it

Requirements

1. Kubernetes Based Deployment

Type	Functional requirement
Priority	Critical

Introduction:

Kubernetes is an open source system for automation of deployment, scaling and management of containerized applications. It groups containers that make up applications to logical units for easy management and discovery.

Aim:

Aim is to convert the existing VM based deployment to Kubernetes based deployment. This will enable CAN applications to be segregated into micro services (logical units) that can be managed using the automation capabilities of Kubernetes and will render superior service using features like load balancing, self-healing, auto scaling etc.

Requirements:

Requirement ID	Requirement Description
REQ05501	Development of Kubernetes based deployment architecture
REQ05502	Creation of POD images for CAN Services
REQ05503	Enable Service Level Load Distribution and Traffic Management
REQ05504	Enable hardware load balancer (i.e, using cloud vendor provided application load balancer) and software load balancer (i.e., using Nginx Web Server)

2. Elasticstack Integration in OpenShift

Type	Business Requirement
Priority	Important

Introduction:

Elastic Stack is a group of open source products from Elastic designed to help users take data from any type of source and in any format and search, analyze, and visualize that data in real time. Kibana is an open-source data visualization and exploration tool used for log and time-series analytics, application monitoring, and operational intelligence use cases

Aim:

Elasticstack Integration with CAN Kubernetes ecosystem allows the Pod logs to be visualized in Kibana.

Requirements:

Requirement ID	Requirement description
REQ05505	Collecting logs from the Pods
REQ05506	Keeping the details of the killed logs
REQ05507	Allowing Pod logs to be visualized in Kibana UI

3. Pod Logging

Type	Non-Functional Requirement
Priority	Important

Introduction:

Pod Logging is particularly useful for debugging problems and monitoring cluster activity.

Aim:

Aim is to enable pod logging in CAN Kubernetes ecosystem.

Requirements:

Requirement ID	Requirement Description
REQ05508	Removal of support for file based logging
REQ05509	Ensuring application logs are available as Pod's container logs

4. Horizontal Pod Autoscaling – HPA

Type	Functional Requirement
Priority	Important

Introduction:

Horizontal Pod Autoscaling is a Kubernetes feature where the pods can auto scale based on CPU utilization or custom metrics.

Aim:

Implementation of Horizontal Pod Auto Scaling in CAN, so that it automatically scales the number of Pods in a replication controller, deployment, replica set or stateful set based on observed CPU utilization.

Requirements:

Requirement ID	Requirement Description
REQ05510	CAN GUI application and Prediction controller modules should support HPA to meet increased demands of GUI requests and Prediction batch requests
REQ05511	Prediction worker modules should automatically scale (upward/downward) to efficiently run the atomic predictions based on the increasing demands from the Prediction controller when the CPU utilization hits a configured threshold
REQ05512	Allows configurations CPU target utilization, minimum and maximum replica count for all the modules that support HPA

5. Helm Charts for Easy Installation

Type	Functional Requirement
Priority	Important

Introduction:

Helm charts helps in management of CAN pods. It helps user to define, install and upgrade Kubernetes application.

Aim:

Aim is to enable Helm chart based implementation for CAN pods/ Kubernetes.

Requirements:

Requirement ID	Requirement Description
REQ05513	Creation of Helm Charts for CAN modules
REQ05514	Manage the release, upgrades and uninstallation of CAN Kubernetes through Helm charts

6. Integration with Istio Service Mesh

Type	Non-Functional Requirement
Priority	Important

Introduction:

Istio is an open source service mesh that layers transparently onto existing distributed applications. Istio's powerful features provide a uniform and more efficient way to secure, connect, and monitor services.

Aim:

Integration of Istio service mesh to CAN ecosystem of Kubernetes based deployment and to enable micro service monitoring besides discovery, load balancing, failure recovery.

Requirements:

6.1. Service Mesh with Envoy proxy

Requirement ID	Requirement Description
REQ05515	Each workload/pod should get deployed along with its own envoy sidecar proxy. These envoy proxies should provide features like traffic management, service authorization, load balancing, dynamic service discovery etc.,

6.2. Authorization Policy

Requirement ID	Requirement Description
REQ05516	Authorization policy is used to control how different modules of CAN applications share data with one another using ALLOW/DENY permissions on all the workloads deployed in the CAN workspace
REQ05517	Authorization policy is also enabled to specify the HTTP methods (GET, POST, PUT etc.,) that have to be used while communicating with other pods inside the namespace

6.3. Peer Authentication Policy

Requirement ID	Requirement Description
REQ05518	It allows to configure all the workloads in CAN workspace to only accept requests encrypted with TLS in STRICT mode

7. Integration with Monitoring Softwares (Kiali, Prometheus and Grafana)

Type	Functional Requirement
Priority	Important

Introduction:

Kiali is the management console for Istio based service mesh. Kiali provides dashboards, observability, and ensure mesh operation with robust configuration and validation capabilities. It shows the structure of service mesh by inferring traffic topology and displays the health of mesh.

Prometheus is a free software application used for event monitoring and alerting. It records real-time metrics in a time series database built using a HTTP pull model, with flexible queries and real-time alerting.

Grafana is a multi-platform open source analytics and interactive visualization web application. It provides charts, graphs, and alerts for the web when connected to supported data sources

Aim:

Integration with Kiali, Prometheus and Grafana to get more insights into performance of containerized applications, Kubernetes clusters, Docker containers, and underlying infrastructure metrics.

Requirements:

7.1. Integration with Kiali

Requirement ID	Requirement Description
REQ05519	Shows the structure of service mesh by inferring traffic topology and displays the health of the service mesh
REQ05520	Facilitates to view the application logs of any interested pods from Kiali dashboard
REQ05521	Facilitates to view pod health, inbound and outbound traffic direction and MTLS configuration

7.2. Integration with Prometheus

Requirement ID	Requirement Description
REQ05522	Collects the real-time metrics in a time series database of all the running microservices within the system. Internally these metrics are used by Grafana and Kiali dashboards

7.3. Integration with Grafana

Requirement ID	Requirement Description
REQ05523	“Avanseus_Dashboard” preloaded with visualization of CPU utilization of database, master node and worker node
REQ05524	“Avanseus_Dashboard” preloaded with visualization of HTTP request/response stats between consumer, controller and worker nodes
REQ05525	Ability to add/modify the dashboard as per the requirements by writing appropriate queries related to microservices data

8. ServiceNow Integration

Type	Business Requirement
Priority	Critical

Introduction:

ServiceNow is an enterprise entity that provides solutions for IT asset management and other digitalization drives that happens in the IT ecosystem. One of the key product of ServiceNow includes the IT Service Management Tool that helps the telecom, IT customers to log in fault incidents, track and close them through the digital work flows.

Aim:

Main objective of integration is to optimize the customer operations. It had been noted that there are multiple customers of CAN using ServiceNow ITSM tools and have raised the concern of integrating the software for seamlessness. This integration will bring in the seamlessness among the operation of both software mutually complimenting the cause of enhancing the customer operations and performance.

Requirements:

Requirement ID	Requirement Description
REQ05526	Creation of ServiceNow connector application and UI support
REQ05527	Real-time extraction and display of predictive tickets with filters in tabular and graphical forms
REQ05528	Creation of single ticket and multiple ticket directly from Prediction Data
REQ05529	Retrieving the data of already open tickets and UI support to update the same including engineer assignment, resolution comment & status

Requirement ID	Requirement Description
REQ05530	Option to close or terminate a predictive ticket
REQ05531	Option to archive, download, save and print reports regarding analysis of predictive tickets

9. Kafka Integration

Type	Functional Requirement
Priority	Important

Introduction:

Apache Kafka is a framework which allows processing of streaming data. It is an open source platform developed by Apache Software Foundation and provides unified, high throughput, low latency platform for handling real time data feeds.

Aim of Integration:

Integration of CAN application with Kafka broker optimizes customer operations of sending the alarm, ticket, and performance counter data in a streaming channel. Earlier the data was being fed to CAN in a traditional flat file format on a daily basis. This streaming interface allows CAN to subscribe to it and digest data in real time.

Requirements:

Requirement ID	Requirement Description
REQ05532	Creation of consumer API
REQ05533	Creation of connector API
REQ05534	Creation of Stream API
REQ05535	Creation of Admin API
REQ05536	Creation of UI support configuration
REQ05537	Audit information of data collected to be shown in Monitoring screen

10. Memcached Tool Integration for Centralized Cache

Type	Non-Functional Requirement
Priority	Important

Introduction:

Memcached is a general purpose distributed memory caching system. It is used to speed up dynamic database driven applications by caching data and objects in a centralized server. This is a free and open source software.

Aim:

Memcached Tool Integration is used for centralized session storage for distributed tomcat setup. It also enables CAN to keep the temporary data or caching data in the centralized server for faster retrieval of data.

Requirements:

Requirement ID	Requirement Description
REQ05538	Integration of Memcached tool in CAN & CAS for session storage
REQ05539	Integration of Memcached tool to store login tickets against the session IDs

11. NFS Server Data Storage for Pods

Type	Non-Functional Requirement
Priority	Important

Introduction:

The Network File System (NFS) is a client/server application that allows a computer user view and optionally store and update files on a remote computer as though they were on the user's own computer. NFS server allows the storage for stateful Pod data.

Aim:

Aim is to enable NFS storage for CAN pods so that data persistence can be enabled where destruction of pods doesn't destroy data. With the feature of data accessibility to multiple pods at the same time, it also allows sharing of data between the pods.

Requirements:

Requirement ID	Requirement Description
REQ05540	Creation and configuration of NFS servers
REQ05541	Use of NFS volumes for pods

12. Mongo DB TLS Integration

Type	Non-Functional Requirement
Priority	Important

Introduction:

Transport Layer Security (TLS) is a cryptographic protocol that enables end-to-end security of data sent over internet. It avoids and prevents possible eavesdropping or alteration of such data being sent ensuring the sanctity of the data.

Aim:

TLS integration in MongoDB is to ensure secured way of data transmission between the MongoDB server and the client application.

Requirements:

Requirement ID	Requirement Description
REQ05542	TLS/SSL enablement over Mongo DB instances
REQ05543	Certificate Management

13. Advanced RoE and Ticket Matching

Type	Functional Requirement
Priority	Critical

Introduction:

Return on Effort (RoE) index based prediction shortlisting is a way to select a particular subset of predicted faults which are more impactful or likely to happen and highlight them in the prediction report. This impact or likelihood of faults are determined by taking cumulative effects as measured by weight indices of different parameters like fault history, ticket history, alarm occurrences, ticket correlation, service impact, rarity etc.

Aim:

Advanced Return on Effort (RoE) provides flexibility to efficiently control the number of predictions to be selected through policy configurations. Now, CAN not only prioritizes predictions based on automatic correlation with tickets available on the history, but it can refine prioritization based on user provided policies too. Apart from the existing parameters provided in previous release, there are new parameters in policy configuration given by default to improve the accuracy of RoE prediction. These are:

- Ticket correlation – To match more reactive tickets
- Service impacting – To match more service impacting alarms
- Rarity – To match rarer alarms
- Prioritized cause category – To match more hardware & Infra alarms than Transmission alarms
- Work order count – To match more field tickets/work orders

Requirements:

Requirement ID	Requirement Description
REQ05544	UI support to define, select and delete RoE policies
REQ05545	Execution of defined policies in post prediction phase

Requirement ID	Requirement Description
REQ05546	UI support for report generation, download, archive etc.

14. Web Security Configuration Porting from Apache to Nginx

Type	Non-Functional Requirement
Priority	Important

Introduction:

Nginx is a high reliable and secure web server that can be hardened further based on user requirement to cater applications of different criticality. This supports open source implementations for popular web server hardening approaches and security standards.

Aim:

To port all configurations of CAN from existing Apache HTTPD server to Nginx web servers. This will bring in more flexibility for managing configuration changes in adhoc customer requirements, Reduced time towards identifying the service affecting activity, overall improvement and efficiency of manpower by reduced waiting time for user, enable on the fly upgrades and load balancing.

Requirements:

Requirement ID	Requirement Description
REQ05547	Creation and configuration of Nginx web server
REQ05548	Porting of previous environment configurations in Apache server to Nginx server configurations
REQ05549	Porting of previous security configurations in Apache server to Nginx server configurations

15. Tomcat Clustering for Session Management in CAN & CAS

Type	Functional Requirement
Priority	Important

Introduction:

Clustering of Tomcat servers enables a group of servers serving the incoming HTTP requests rather than single server doing it. This enables high availability for the HTTP requests even if one or few tomcats are down in the tomcat cluster.

Aim:

The CAN and CAS tomcat applications are clustered, which enable them to run in multiple instances for high availability. This will enable better HTTP request load management and session recovery in case of server crash as session IDs will be shared among the cluster members.

Requirements:

Requirement ID	Requirement Description
REQ05550	Configuration of Tomcat clustering for CAN & CAS
REQ05551	Compatibility of clustering with Kubernetes horizontal pod auto scaling

16. UI Porting with React Framework

Type	Non-Functional Requirement
Priority	Important

Introduction:

React/React JS is open source front end java script library used for building UI components. Such UI components together will constitute the complex UI of application improving overall dashboard experience.

Aim:

UI Porting with React Framework helps to increase the performance of the application. This integration separates the front end module from backend services & communication between the front end & backend module happens over REST API.

Requirements:

Requirement ID	Requirement Description
REQ05552	Creation of UI components in React JS
REQ05553	Configuration and integration of UI components
REQ05554	Integration of Spring REST API for create, update, view or delete operations with the backend

17. Mongo DB Sharding

Type	Non-Functional Requirement
Priority	Important

Introduction:

Sharding is method of distributing data across multiple machines. Mongo DB uses sharding to support deployment with very large data sets and high throughput operations.

Aim:

Mongo DB sharding enables us to handle large amount of CAN data. This implementation will improve the efficiency of data processing due to horizontal scaling, reduce overall cost of implementation and overall better management of work load.

Requirements:

Requirement ID	Requirement Description
REQ05555	Creation of shard cluster in Mongo DB
REQ05556	Configuration of cluster for CAN operations

18. Mongo DB Version Upgrade from V3.4.6 to V4.4.5

Type	Non-Functional Requirement
Priority	Important

Introduction:

Upgrading to the latest version of Mongo DB as the version 3.4.6 is out of support.

Aim of Integration:

Aim is to upgrade the Mongo DB to latest stable version (v4.4.5) which provides the best support for the CAN database management. MongoDB 4.4.5 is a database designed for ease of development and scaling. Upgradation provide security patches, bug fixes, and new or changed features that generally do not contain any backward breaking changes.

Requirements:

Requirement ID	Requirement Description
REQ05557	Upgrade of Mongo DB and associated modules
REQ05558	Inconsistent features from previous version are discontinued

19. Single Login Session for a User

Type	Functional Requirement
Priority	Important

Introduction:

Management of user log in session to avoid extended vulnerability.

Aim:

Aim is to enable single login session for a user account at an instance and eliminate old login sessions for the same user ID.

Requirements:

Requirement ID	Requirement Description
REQ05559	Allows and allots new user session for a logged in user & logs out or kicks out the session allocated for same user identifier who logged in earlier from another location/browser.

20. Workorder Integration

Type	Functional Requirement
Priority	Important

Introduction:

Predictive tickets/workorder ingestion in CAN allows the software to map it directly to Predicted faults to check what was the action taken on field, time taken etc.,

Aim:

Aim is to implement work order parsing as part of input parsing over user interface. This involves mapping of raw work order data fields with CAN work order fields.

Requirements:

Requirement ID	Requirement Description
REQ05560	UI support for parsing workorder details

21. Parser UI Enhancement

Type	Functional Requirement
Priority	Important

Introduction:

Enhancement to meet end user experience in Parser screen.

Aim:

Aim is to allow users to add multiple columns for parser configuration.

Requirement ID	Requirement Description
REQ05561	Enable multi-selection of columns for a parser in Parser configuration screen

22. Java Security Manager Enhancement

Type	Non-Functional Requirement
Priority	Important

Introduction:

Java Security Manager defines security policy for the CAN application limiting the actions allowed by the users. The Java Security Manager provides a facility to prevent untrusted code from accessing files on the local file system, connecting to a different host, executing harmful commands and many additional restrictions.

Aim:

Aim is to integrate Java security manager preventing users from running scripts that can compromise the CAN application at any level. This is critical as CAN provides additional flexibility to its users by providing options to run code snippets to customize the way CAN has to convert the data. Example: Data parsing, Excel report cell information etc.,

Requirements:

Requirement ID	Requirement Description
REQ05562	Enables java security manager to pick-up configurations automatically
REQ05563	Prevents untrusted code from accessing or modifying system resources
REQ05564	Uses configuration file (security, Policy) to check the permissions during run time and only the configured permissions in the policy file are allowed

23. Re-Scheduling of Trigger based on UI Cron Pattern Configurations

Type	Functional Requirement
Priority	Important

Introduction:

Cron pattern configurations is available across many modules in CAN. They deal with scheduling of job/activity associated with that module. Re-scheduling option enables job to be rescheduled based on the newly configured Cron pattern.

Aim:

Aim is to reschedule the Cron job instantly whenever it is updated in the UI. This also ensures that the job gets schedules in one of the tomcats in clustered setup. If any of the tomcats goes down, the scheduler runs the job in a tomcat that is alive at that moment.

Requirements:

Requirement ID	Requirement Description
REQ05565	Reschedule Cron jobs to an appropriate time instantly whenever it is updated from the UI

24. Prediction as a Service

Type	Functional Requirement
Priority	Important

Introduction:

Prediction as a service architecture decomposes the existing prediction flow architecture into 3 components:

a. Consumer: Responsible for generation of prediction input
b. Controller: Responsible for collecting the prediction input & calling the work service in batch mode
c. Worker: Responsible for performing atomic prediction

Aim:

Aim of Prediction as a service is to allow each component mentioned above to perform its tasks independently & also allows components to remain completely autonomous and unaware of each other.

Requirements:

Requirement ID	Requirement Description
REQ05566	Worker Service: Ability to call the worker service to get instantaneous result of a given prediction input sequence.
REQ05567	Controller Service: Ability to upload the prediction input file using multipart request and download the entire output file.

25. REST API for prediction delivery

Type	Functional Requirement
Priority	Important

Introduction:

Prediction delivery in the previous releases used to happen via Email excel attachment or customer had a provision to download the same Excel report from CAN Dashboard. With the implementation of REST API for prediction delivery, customers can download the prediction report for a desired date over a REST call.

Aim:

Aim of REST API for prediction delivery is to allow third party customer application to directly download the prediction report over REST call & integrate it with their ticketing system. The REST API is documented using Swagger API specifications.

Requirements:

Requirement ID	Requirement Description
REQ05568	REST API delivers prediction report in JSON format
REQ05569	REST API documentation using Swagger specifications defining the REST URL, input format & output format
REQ05570	REST API support for username & password authentication over HTTP Auth header to authorize access to the REST API URL