CAN 5.5

Release Document

JULY 6, 2021

AVANSEUS TECHNOLOGUES PVT. LTD.

Table of Contents

General Remarks

CAN 5.5 is one of the intermediate releases for 2021. At the discretion of delivery team, deployments using CAN 5.0 can be upgraded to CAN 5.5, based on each customer situation, to get the new features & also to run CAN in Kubernetes based setup. All new deployments must use CAN 5.5.

Please review these release notes to learn what is new in this version as well as important notes concerning known issues, bug fixes for 5.0 and improvements in the existing features.

New Features

- **User interface recoded in React** makes the UI application to be completely decoupled from the backend server module. Communication between UI & Backend server module happens via REST. The UI allows users to navigate across different screens very quickly and has a consistent high responsive interface that give quick outputs. UI also has been developed with React native component libraries which is both captivating and engaging.
- **Kubernetes based dynamic and scalable deployment** enables CAN compatible to cloud native deployment environments. CAN is compatible to docker which enables automation of deploying, scaling and management through various orchestration applications like Kubernetes. With CAN 5.5, the deployment of entire CAN tool package is moved from ***VM based setup to Kubernetes based***.
- **Elasticstack integration to collect Pod logs** allows users take data from any type of source and in any format and search, analyse, and visualize that data in real time in a tool named Kibana. With Kibana, users are allowed to view & analyse even the deleted Pod logs.
- **Horizontal Pod Autoscaling** allows CAN submodules like GUI & Prediction worker application to replicate/scale the application automatically based on CPU load. This integration ensures high availability of the GUI and prediction worker application.
- **Helm charts for application installation** enables the deployment team to install, uninstall & upgrade the application very easily compared to installation of application in a VM based setup which needs many manual activities.
- **Integration with Istio service mesh** enables micro service management of CAN application pods besides discovery, load balancing & failure recovery. This also allows deployment team to configure Istio gateway & an Ingress controller in Kubernetes which is helpful in exposing the CAN application GUI over HTTP/HTTPS channel.
- **Integration with monitoring tools (Kiali, Prometheus & Grafana)** : ***Kiali*** is a management console for Istio service mesh tool which provides dashboards, observability, and ensure mesh operation with robust configuration and validation capabilities. ***Prometheus*** is a tool for event monitoring and alerting which records real time metrics in the database & helps monitor the application in run time. ***Grafana*** is a tool which allows interactive web visualization of applications by showing application statistics in the form of graphs, charts & alerts.
- **Service Now integration** enables customer operation optimization as most of the customers using CAN are having ServiceNow ITSM tool in their environment to log tickets. This integration brings seamlessness among both the software mutually.
- **Kafka integration** with CAN optimizes customer operations of sending Alarms, Tickets & Performance counter data in a streaming channel instead of traditional flat file based upload. This integration allows CAN to ingest data in real-time.
- **NFS server data storage for Pods** 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.

- **Advanced Return on Effort (RoE)** is a way selecting a subset of predictions based on index based shortlisting of predicted faults which are more impactful or more likely to happen and highlighting them in the report. The selection of predictions is based on policy configurations where CAN not only prioritizes the predictions based on default policies, but it can refine the prioritization based on user driven policies too. Apart from the default policies given in the previous release for few parameters, there are new parameters considered in the policy which improves the quality of short listing. These are reactive ticket correlation, service impacting alarms, rarity of alarms, Hardware & Infra alarms correlation & field ticket work order.
- **Tomcat clustering for session management** in CAN & CAS enables 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. In combination with Horizontal Pod Autoscaling, these tomcat container applications increase or decrease their replication based on input CPU load.
- **Workorder integration in Parser** screen enables CAN to map the workorder data directly to Predicted faults to check what was the action taken on field, time taken etc., This integration allows 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.
- **Prediction as a service** is a architecture level change from previous releases, where the prediction architecture is broken down into 3 components: Consumer, Controller & Worker. This architecture allows each component to work independently & also allows components to remain completely autonomous and unaware of each other. This is a stepping stone for components like controller & worker to be consumed by other consumers through an API call to run predictions.

Enhancements

- Prediction code base is moved from Java to C++ with vectorization, aligned memory access and memory adjacency for even faster prediction generation.
- Logs of the applications are available as part of Pod's container logs. File based logging is removed
- Memcached tool integration for centralized cache to store temporary data. Session information is stored when the CAN & CAS application is clustered. Also, other information like Login tickets against session IDs is also stored here.
- MongoDB integration with TLS enables data transmission between the applications and database to happen over a secured channel.
- Web security & configurations from Apache HTTPD server is being moved to NGINX server setup.
- MongoDB 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.
- MongoDB version upgrade from v3.4.6 to v4.4.5 is being made both at code & DB collection level as few of the features from v3.4.6 are discontinued & customer environments demand us to go with latest version due to better security & performance.
- Single login session for a user makes sure that there is only one session ID enabled for a user ID at any point in time. This forces out user logged in with same user ID from another browser/location to logout.
- Parser UI enhancement to allow multiple selection of CAN fields at a time & provides users a better experience during configuration.

- Java security manager is now slightly changed to include security policy file as part of CAN build instead of file being configured outside the build.
- Rescheduling of Triggers based on UI cron patterns is supported now in both single & cluster mode. The state of the triggers is being stored in the database.
- Web application security features:
    1. Compliance towards prevention of injection attacks and other vulnerabilities
    2. Compliance of authentication management to prevent unauthorized user access
    3. Protection of sensitive data and prevention of information leakage
    4. Processing of external XML entities
    5. Enhanced access control measures based on roles
    6. Security misconfiguration issues and error handling
    7. Mitigation of cross site scripting related vulnerabilities
    8. Prevent insecure deserialization
    9. Compliance while using components with known vulnerabilities
    10. Compliance in cases insufficient logging and monitoring

Removed Features

- Prediction assignment policy screen is removed as “Prediction as a service” feature follows a seamless multi-level module architecture for Prediction.

Bug Fixes

- MongoDB version upgrade from v3.4.6 to v4.4.5 involved few corrections and fixes at code level as few older MongoDB APIs are discontinued. Code fixes are made to handle aggregate functions & also few changes were done in DB structure level for few collections which were incorrect in earlier versions & now corrected in the newer MongoDB Java driver.

Platform Support

- **Client OS & browsers supported**:

OS	Browsers supported
Microsoft Windows 7 & 8	Google chrome 60.0 and above Mozilla Firefox 66.0 and above
Microsoft Windows 10	Google chrome 60.0 and above Mozilla Firefox 66.0 and above Internet Explorer 11 and above
Apple MacOS 10 and above	Google chrome 60.0 and above Mozilla Firefox 66.0 and above
Ubuntu Linux 15.0 and above	Google chrome 60.0 and above Mozilla Firefox 66.0 and above
Fedora Linux 29.0 and above	Google chrome 60.0 and above Mozilla Firefox 66.0 and above

- **Kubernetes server**: v1.21 and above

Known Issues

- Although CAN has been configured to support internationalization, it is currently available only in English and the CAN binary is packed with English text only. When there is a requirement from the Customer to port the application to support some specific language, then it becomes the responsibility of the delivery side developer to map the translations into a file and then pack the binary with that language.