The Select Group – Utilities Projects

Application Development / System Integration

We developed an application for a customer installing their initial AMI solution, which processes Outage and Restoration messages to filter momentary outages (defined in configuration within the application), in order to reduce the number of Positive Outage Notifications (PON) and Positive Restoration Notifications (PRN) which are sent to the Outage Management System (OMS). The application developed was called the Outage Filtering Application (OFA).

OFA subscribed to the AMI head-end for Outage and Restoration alarms. OFA would suppress the PONs and PRNs from a given meter if the restoration alarms were received within a configurable amount of time after the initial outage alarm. The configurable times were based on reclosure and FLISR operation times. OFA also maintained a list of meters for which there were planned outages as an exception list. A REST interfaced was developed that allowed the workforce management system and meter installation application to set the exception meters within the application. PONs/PRNs were suppressed for the exception list. OFA logged to a local database all incoming operations (outage and restoration reception) and outgoing operations (OMS web service attempts or deliveries). Statistical reporting was developed to provide both a global and device view of system outages. The application was later extended to include assessing meter momentary processing and provide comprehensive statistics on software and meter momentary information for selected meters over a given timeframe. OFA was developed in C# and provided as an on-premises application, including an installer and all necessary documentation. SQL Server was used as the data store.

Cloud-based Headend Development

An example of our cloud development experience is a headend solution operating an IoT sensor network. The application developed provided reading, management, and performance analytics for the sensor networks. A collection of sensors was geographically dispersed at each customer site. The number of sensors in each network ranged from 5,000 to greater than 100,000. The network substrate for communication varied between cellular, 900MHz/2.4GHz mesh and LPWAN technologies. The application was built on the Microsoft Azure platform and hosted on Azure. It was developed as a multi-tenant application with proper security measures in place to ensure data isolation between customers. The application was architected and developed as a collection of micro and macro-services built on .NET Core and C#. The front-end client utilized Typescript and Angular.

The microservices were built to run in Docker containers with each container being able to run on Windows or Linux. For the service bus, Kafka was used for streaming and Azure Service Bus (AMQP) for all other backend communication. For logging the ELK (Elasticsearch, Logstash, Kibana) stack was implemented. Microservice and application monitoring metrics were provided using Prometheus and Grafana, while tracing between microservices was performed using Zipkin. Apache Solr was used as the search infrastructure for the application, and Kubernetes provided the infrastructure for container orchestration. A collection of database technologies was used in the solution, including SQL Server for relational data, Azure Cosmos DB for unstructured data, and Azure Table Storage for reading data. Azure Blob storage was also incorporated for larger unstructured data.

SCADA Migration

This utility had been using RT SCADA for its distribution system for many years. It was in the process of installing an ADMS system as part of its Grid Modernization program. To get the ADMS system properly configured, we were brought in to help in building the overall transformation process from RT SCADA to ADMS. The RT SCADA system is a Microsoft Access application distributed across >150 sites. These distributed DBs needed to be consolidated to perform analysis of the state of the overall system. A data mapping application as built in C# which performs a bulk migration and incremental migration weekly into a consolidated SQL Server database. A SQL Server ETL process was then developed to normalize the telemetry information and then to construct the necessary file imports for the ADMS. We also built a process to fingerprint the various field devices from a point perspective so that we could incorporate Templates within the transformation. Oracle GIS information was fused with the transformation information to provide a comprehensive view of the devices to the ADMS system on import. To standardize device and point names into a new utility standardized approach, a text-based rules engine was developed and used to ensure that names were standardized before being imported into the new ADMS. The consolidation process ran periodically, and differentials stored and reported on. The overall process was developed so that migration could occur for 5 or so substations and subsequent feeders at a time, since the system must continue to be operated in production during the migration.

Application/Database Performance Tuning

A third-party performed a port for Itron's Meter Data Management (MDM) platform IEE, from Oracle to SAP HANA. Following the port, we worked with SAP through a grant from Intel, to test the IEE-HANA implementation in a production environment. The utility engaged in the project uses IEE for collecting and processing data from its Advanced Metering Infrastructure (AMI) for both Electric and Gas meters. A subset of meters from the overall population was published to the SAP Co-Innovation Lab (COIL) to perform real-world analysis of the IEE-HANA solution. Standard operational aspects of IEE like Validation, Editing and Estimation (VEE) rules and the configurations for the subset of meters from the utilities production system were implemented in the COIL environment. Several weeks were spent to ensure that the COIL system configuration was correct and processing data accurately. Following verification, the utility began publishing production data from the meter population.

The database and application were analyzed for bottlenecks and for any limitations within the solution. During these tests, we implemented several changes to the database procedures (procs) and configuration of IEE to better align and more effectively use the processing power of the HANA database. The team was able to improve the performance of the system by a factor of 7x and identified several areas both in database procedures and application code where changes could provide further improvements to the system. The outcome of the project demonstrated that HANA could provide a stable platform for running IEE and that the HANA platform can provide improvements in processing speed for inbound data, allowing for new real-time analytics to be developed which can provide more operational insight and capabilities for utilities than the currently supported databases on IEE.