EDGE ANALYTICS FOR THE INDUSTRIAL IoT

A GEOGRAPHICALLY DISTRIBUTED ANALYTICAL ARCHITECTURE

OVERVIEW

This paper considers the reasons for using a geographically distributed analytical architecture in an Industrial Internet of Things (IIoT) context.

A typical IIoT architecture performs event processing (CEP) and filtering near the source of data and ships the outcomes of event processing into the cloud for alerting and analytical purposes. Much of the granular source data is lost because network bandwidth limitations make the transport of voluminous operational data into the cloud impractical.

However, this architecture is reminiscent of a legacy data warehouse system where operational systems feed a limited synopsis of operational activities into a central warehouse and the partial view of the operation history ultimately limits the analytics possible in the future.

Contemporary analytical systems take an approach whereby the granular operation detail is retained in an exhaustive long-term data manner to allow arbitrary exploration and analysis of any aspect of operations now and in the future. This is especially relevant when business environments are dynamic, and it is not possible to predict which queries/responses will deliver the greatest insight.

Providing the retention of everything will afford the analysis of anything. The potential value of any given datum remains unknown until some point in the future and that value might only be realized when combined with other data.

There is also some dissonance between the current architecture for IIoT analytics and contemporary wisdom which promotes pushing the analytical processing to the data rather than moving data to an analytical engine. The reasoning for moving processing to the data partly arises from the limitations of network bandwidth which presents a barrier to volume scaling; and in a wide geographically distributed network, those limitations become even more evident.

EDGE STORAGE

Storing and analyzing operational data at the edge of the network affords a number of advantages over a solely centralized cloud analytical approach.

Leaving the data at the edge avoids the bandwidth limitations of geographically distributed networks and obviates any need to filter out granular operational detail. Filtering data forces premature decisions about which data is shipped and which is discarded and those decisions are inevitably based on incomplete information about the potential value of the data being discarded. Whereas, collecting and retaining data at the edge allows everything to be retained for potential analysis at some point in the future.

Even with an architecture of filtering and shipping data to a central cloud, there is still a requirement to store and forward data when network connectivity fails and data cannot be immediately transported away from an edge point - so retaining data at the edge indefinitely can be viewed as an extension of an existing store and forward requirement.

CLOUDLETS

An architecture where compute resources are located close to mobile devices to overcome network latency and jitter for compute intensive mobile interactions has been proposed for mobile communications networks (http://research.microsoft.com/en-us/um/people/bahl/Papers/Pdf/cloudlets09.pdf). These compute resources are distributed around the network, are stateless and provide compute services on demand to mobile devices located nearby. In this architecture, these local resource centers are called Cloudlets.

It is reasonable to extend this architecture to include data storage in the Cloudlet, especially as Cloudlet services are likely to produce voluminous operational data from their activities which presents a useful source for analytics.

When used in an IIoT context, these Cloudlets would sit between the device gateways and the cloud and operate as data storage hubs to nearby gateways. In this architecture, CEP continues to operate in the device gateways, but instead of discarding the operational detail that has been processed, the detail is passed through to its nearby Cloudlet.

With the operational data retained within the Cloudlets, analytical processing is pushed from the cloud down to the Cloudlets and their results aggregated in the cloud. This offers a number of benefits for analytical queries:

- Cloudlets provide comprehensive, geographically distributed edge analytics for arbitrary data exploration and analysis.
- Analytical processing can be pushed into the Cloudlets to bring the analytical processing closer to the data while avoiding the transport of large amounts of data over geographically distributed networks.
- Analytical processing pushed into a Cloudlet can exploit the resources available in the Cloudlet and queries can operate in parallel across multiple Cloudlets.

- Cloudlets can provide high-availability for data collection and queries by replicating data between nearby Cloudlets. Queries can operate in parallel across the available Cloudlets at the time of the query and can load balance between Cloudlets that contain common data.

Moreover, real-time stream processing can also operate in the Cloudlet to provide real-time alerts to anomalous conditions that might arise in the context of multiple gateways.

This is not to say that data should never pass beyond a Cloudlet into a cloud, but rather that a hybrid architecture overcomes the limitations of pushing everything into the cloud and makes a comprehensive geographically distributed edge analytics readily feasible.

For the purposes of this paper, we will repurpose the use of the term Cloudlet to a Data Store, beyond the use within mobile networks that is described above.

DISTRIBUTED QUERIES

The majority of analytical queries are likely to be submitted from the cloud and may address one, many or all of the Data Stores in the network. Performing a query across multiple Cloudlets requires the query to be disseminated and distributed across all of the Data Stores with the heavy lifting pushed down into the Data Stores to avoid excessive data transport back to the point of query. This involves performing the majority of the predicate filtering, join, aggregation, grouping and ordering processing within the Data Stores and merging their results into a combined result.

In the context of a relational query against structured data, distributing the query processing involves well understood techniques provided that:

- A common schema is applied in every Data Store.
- Universal data which is not specific to any particular Data Store is replicated across all Data Stores.

The former requirement is a natural extension from using a single schema in the cloud; while the latter dictates certain aspects of how data must be distributed and/or replicated across Data Stores.

Where queries are distributed across Data Stores, they can be submitted and executed in parallel to minimize overall query execution time. Where common data exists in more than one Data Store, the currently available Data Stores can be chosen at query time to provide high-availability; and where one Data Store is required from a choice of multiple currently available Data Stores, that choice can be random or based on current Data Store utilization to offer load-balancing across them.

ORCHESTRATION

The orchestration of operational activities becomes more complex with geographically distributed Data Stores, largely because of the requirement to guarantee the roll-out of changes across a high-latency low-bandwidth network which is likely to suffer partial loss of connectivity at any time.

An essential user requirement for managing a complex system is the provision of a single pane of glass management console which provides a unified view of the system as a whole. Operational changes need to be instructed precisely once and fulfilled in an autonomous and asynchronous manner across the network of Data Stores.

This requires a reference source in the cloud which immediately evaluates every request and journals the fulfilment instructions to be propagated out to every Data Store as and when there are available. The reference source must be distributed across multiple servers and employ quorum semantics with transactional atomicity and durability to guarantee availability and immediate consistency within the reference source itself.

Meanwhile, eventual consistency has to be applied to every Data Store to mitigate against connectivity and hardware failures. Whenever a Data Store has visibility of the reference source it is required to synchronize its state with that of the reference source – which is always consistent.

AUTONOMY

Many database systems rely on either human design effort and/or require large amounts of hardware resource to achieve performance. The challenge for Data Stores is that neither of these will be available near the edge of the network. To put this is some context, it is reasonable to expect each Data Store to be storing tens to hundreds of terabytes of operational data.

Data Stores must run autonomously without requiring regular maintenance, capacity planning, tuning or optimization to reach performance requirements. This autonomy is especially important as Data Stores will vary in both data volume and data distribution and it is not reasonable to expect each Data Store to be managed individually.

Moreover, if the virtues are to be fully realized, then the analytical queries submitted will be unknown and varied and Data Stores will be required to serve them in a responsive manner. This becomes even more essential for Data Stores operating at the edge for many years as requirements inevitably change over time.

Autonomy is also an important virtue for a distributed query strategy, as it permits the bulk of the heavy lifting processing to be pushed out to the edge of the network. Without autonomy, limitations have to be placed on what workload can be pushed out and concerns arise over the ability of all of the Data Stores to respond quickly; and the overall query response time will be largely dictated by the slowest Data Store.

SECURITY

Data Stores will operate in an industrial and potentially hostile environment. The data hosted in each Data Store must be held securely despite operating outside the conventional secure environment of a cloud data center.

A number of specific security points arise out of this:

- Direct query access to a Data Store from resources external to the network must be prohibited.
- Data transported over networks from gateways to Data Stores and from Data Stores to the cloud must be encrypted during transit.
- Ideally, data at rest in the Data Store is also encrypted to guard against hardware theft and physical intrusion into the Data Store.

In addition to protection from external parties, the visibility of data to queries should be managed in a framework of roles and privileges to guard against accidental exposure of sensitive data to legitimate users.

SUMMARY

An architecture of widely distributed Data Stores brings numerous analytical virtues to the IIoT. The architecture can be viewed as federated, geographically distributed edge analytics which offers numerous benefits to IIoT and beyond.

But some essential qualities are required to implement effectively:

- Data Stores must be able to operate autonomously.
- The population of Data Stores must be orchestrated and queried centrally as though operating within a cloud data centre.
- The deployment of Data Stores outside of secure cloud data centres places critical security demands on the systems operating within the Data Stores.