Understanding our business with app telemetry

Microsoft IT supports a wide variety of apps and services for Microsoft Global Sales, Marketing, and Operations (GSMO). GSMO uses these apps to engage with customers, track leads, fulfill goals and objectives, and deliver products and services. GSMO apps come from several different sources: Microsoft product group development teams, third-party vendors, and our development team at Microsoft IT. As the reach and responsibility of GSMO increases, so does the need for a more complete and concise picture of our end-to-end business process.

We moved or refactored most apps in the organization for our cloud-first, mobile-first vision using Microsoft Azure. The cloud is our default target for new apps in development. Azure increases the reach and mobility of GSMO, and it provides an app experience that is resilient, scalable, and consistent.

Looking to serve the needs of the customer

Most apps in the GSMO app portfolio have built-in logging and reporting, but because the sales process extends across many apps, data sources, and environments, we wanted a standardized method for collecting data. Once data is collected in a central location, we aggregate it, combine it, and manipulate it to gain insights into user behavior and the end-to-end sales process.

We worked with GSMO to determine how a telemetry-based data solution could address several issues that their employees were experiencing:

- GSMO wanted a better understanding of the information that is available, and they needed to avoid duplicating data or work. Employees use a wide variety of apps, including some with overlapping functionality, depending on the customer and product they work with. It was difficult to get 100 percent of the information about the sales process without moving in and out of several different apps and reporting systems.
- There were multiple apps, and most of them were in the cloud or moving to the cloud. These apps gathered information using different standards and taxonomies, which made it difficult to compare and combine data from them in a meaningful way.
- We wanted to extend monitoring and data collection methods to include Azure-specific information and methods that were becoming more and more prevalent in the app portfolio as we migrated to Azure.
- We needed to track the health of the GSMO app portfolio in a holistic manner, rather than on an app-by-app basis. We wanted to see and understand the health of the complete sales process as an opportunity moves from app to app. For example, when a sales person looked at an open deal opportunity in Microsoft Dynamics CRM Online, they weren’t connected to the customer details in the Customer Planning app, so they couldn’t see how this deal would affect the overall GSMO team’s goals or take any actions connected to the opportunity.

Creating a framework for telemetry

Telemetry was the first step in the journey to know the customer better. We understood that one of the most important factors in bringing telemetry data together from multiple sources was developing a taxonomy to identify and label data across multiple systems. Our telemetry taxonomy is composed of three schemas:

- **Part A: System.** These fields are defined by and automatically populated by the Logging Library on the local system where events are produced. There might be some fields that the Logging Library would need to get from the caller, but most of the time the values populate automatically. Examples include Users, ClientDeviceIp, and EventDateTime.
- **Part B: Domain Specific.** The different Part B schemas and the fields they contain are defined by centralized groups; the event fields are populated by code written by the Event Author. The Event Author has no control over the field naming or data types. Examples include PageView and IncomingServiceRequest. These are reviewed and published by the Microsoft Data and Analytics team.
- **Part C: Custom Schemas.** These fields are defined by the Event Author, who has complete control over naming and data type of the fields. The Event Author is responsible for populating these fields when an event is instantiated. Examples include OrderNumber and ProductSKU.

The most important aspect of extending the telemetry environment across apps was establishing a common identifier for customers and transactions that existed in all apps. We developed a set of extensions that enabled the creation and maintenance of a common identifier we refer to as a correlation ID. The correlation ID allows us to pass information between apps for an employee or transaction so that data pulled from applications can be organized and displayed by that ID.

Extending apps for telemetry

To capture data effectively using the prescribed taxonomy, we created extensions to integrate into apps that were being developed. These extensions allowed us to set standards for telemetry data across multiple apps, which made it easier to query data and present it from the perspective of the customer experience. The extensions were created to be business-group agnostic. We have used them for sales apps, but they can be integrated into any app.

- **App Insight Extensions.** Provides a standard way of propagating the correlation ID across different services in order to trace business processes that span different service boundaries.
- **Web Extensions.** Provides templates to trace business process events and feature usage events in a standardized way for web applications.
- .**JavaScript Extensions.** Provides templates to trace business process events and feature usage events in a standardized way for JavaScript.

For apps that can’t be extended, logging and telemetry data (log files, database info, and other data sources) are ingested from the application source location. This process is managed and executed using Azure Data Factory to automate the ingestion process on an app-by-app basis.

Building telemetry in Azure

Most of the app and process infrastructure is hosted in Microsoft Azure, so we knew that we wanted a solution that was based in Azure as well. Azure gives us the advantage of being instantly resilient, scalable, and globally available, and it has several components that we were able to use immediately in our telemetry solution.

Application Insights

Application Insights gives us the ability to monitor sales apps to help us detect and diagnose performance issues and retrieve the most important telemetry data from the sales environment. With Application Insights, we can analyze usage patterns and detect and diagnose performance issues within the sales app stack.

Azure Data Factory

Azure Data Factory (ADF) is used to move and transform data. ADF makes it simple to move data between different sources of telemetry data and the central telemetry repository in Azure Data Lake (ADL). We use ADF to transform and analyze incoming telemetry data (from Application Insights blob storage, custom SQL logs, etc.) in order to prepare it for processing by ADL and consumption by reporting.

Azure Data Lake

ADL is the main repository for telemetry data of any size, shape, or form. ADL simplifies the process of ingesting and storing telemetry data and makes it easier to provide accurate data representation to our analytics processes. We use Azure Data Lake Analytics U-SQL scripts to identify user sessions across the massive telemetry data sets.

Creating the telemetry dataflow

The sales telemetry architecture includes the following components and steps that help to collect and present telemetry data for the sales environment:

1. The telemetry extensions are built into apps or used to mine data from apps that don't support the extension.
2. The data from apps is pulled into ADF as raw data. ADF passes the data into ADL.
3. In ADL, raw data is converted and transformed using U-SQL, the native query language for ADL, and put into common schema outputs and aggregation outputs.
4. The data from these outputs is presented using U-SQL for consumption by reporting and visualization tools like Power Query or Power BI.

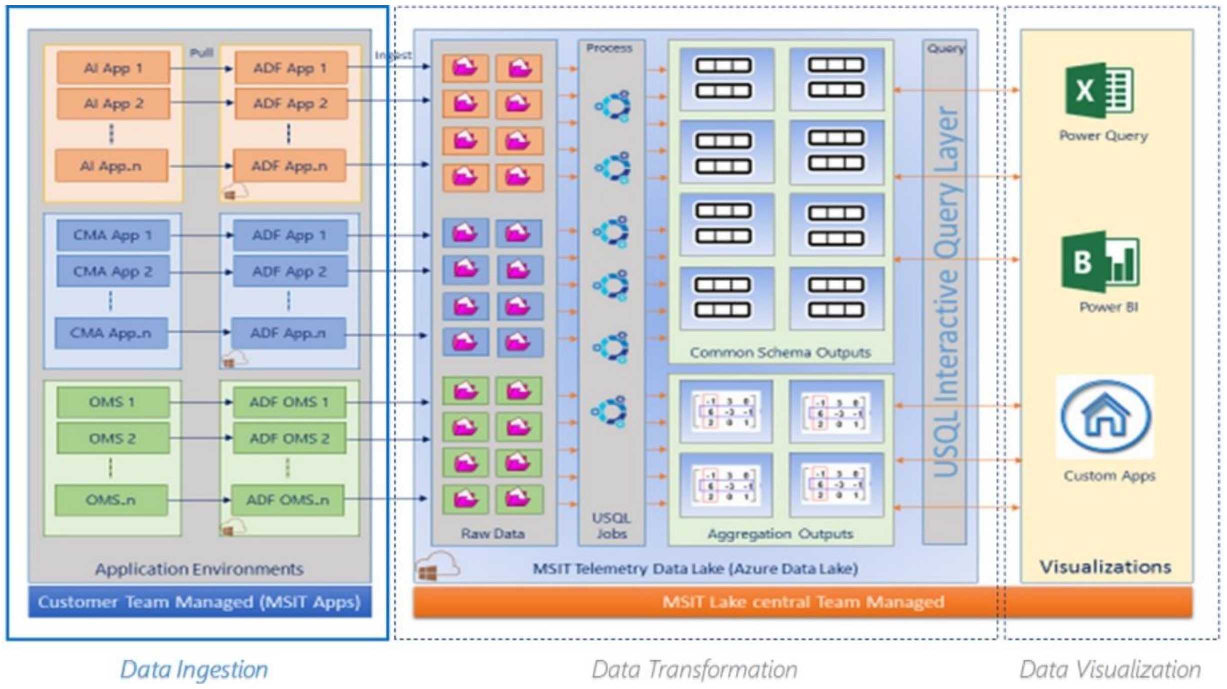

Figure 1. Telemetry architecture

Providing accessible and meaningful data

We used several solutions to provide meaningful results to the GSMO team. Azure App Insights telemetry aggregation with Power BI dashboards are the most common way for GSMO employees to gain insight into their environment. With Power BI, we are able to create visualizations to represent data and trends in ways that were previously unavailable to the GSMO team. For example, Figure 2 is a chart that shows data flowing between several GSMO apps. The dataflow extends beyond a simple app-to-app relationship to encompass the larger business environment and the thirteen apps represented on the chart. Visualizations like this help the GSMO team to better understand some of the underlying behaviors and trends that affect their business.

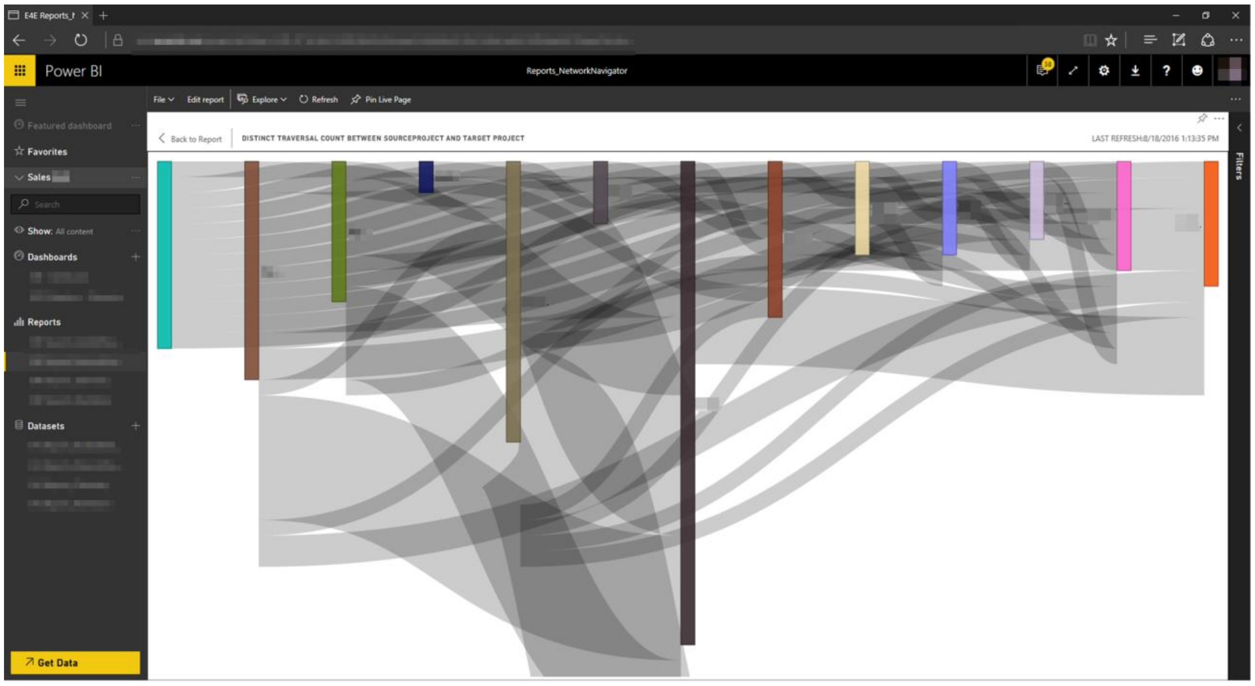

Figure 2. An example of a telemetry dashboard in Power BI

Establishing a customer-focused culture

Traditional IT has typically focused on making technology work, while business teams did their best to work with the available tools and provide as much value as possible to their customers. We found that with Azure, the extra time and effort saved by not having to deploy infrastructure and manage a traditional datacenter allowed us to dedicate more resources to innovating and improving app development. We were able to gain immediate business insight using built-in Azure tools and dashboards instead of building them first.

These changes to the development process taught us that our engineers could and should approach app development from the perspective of a customer. Our tools help sales people to know their customers better—and we learned more about the business and how to make customer-focused decisions during the development process.

What we learned

We established several best practices while we developed the telemetry solution, including:

- Scalable data storage is key. With telemetry, we are collecting massive amounts of data; some will be queried immediately and some may be queried less often or possibly not at all. Regardless of how the data is used, we needed to create a scalable data storage solution to account for the large influx of data.
- A common schema is important. Consistent taxonomy makes it much easier to correlate data between apps and establish a consistent telemetry environment that provides a complete picture of business data.
  However, developing the common schema should not supersede data collection. You can begin ingesting whatever telemetry/logging data you have, even if you do not have a common schema. If the data is there, you can always transform it later.
- Identify the insights you want to get, then build the visualization. Practical business application is important. Don’t let the format and organization of your data dictate the insights you gain from it. Decide which business insights you want to expose and transform your data and telemetry collection accordingly.

A reusable telemetry solution

Our telemetry solution for the GSMO team has provided new insights into how we run our business. Using a common schema and telemetry extensions has allowed us to bridge the gap between sales applications and gain a better perspective on our end-to-end sales process. Our sales people are better informed and equipped for their jobs, and we have developed a reusable telemetry solution that can be extended to other parts of our business.

For more information

Microsoft IT

http://www.Microsoft.com/ITShowcase