***Limitations Maintaining Data Integrity in Azure -* Technological Uncertainties** (5*0 lines of 78 characters)*

The goal was to modify the our application to work with Azure’s Platform as a Service (PaaS). The app was originally custom built by US to work with Amazon Web Services, but in order to save operational costs, it was proposed that the migration to the Azure platform would rely on built-in Azure PaaS services. However, the functionality of the built-in services was limited. For example, it was not known how the database could be migrated while maintaining time zone information. Also, Azure did not provide a cost-effective method for scheduling and managing jobs. Experimental development would be required to overcome the technological issues encountered during the Azure migration for which there were no known solutions.

**Work Done *(100 lines of 78 characters)***

Database Migration

The database migration was constrained to SQLAzure because VMs or managed instances could not be used. Microsoft’s data migration assistant was used to identify functionality that needed to be migrated as part of the compatibility upgrade but could only identify about 70% of the issues. Systematic analysis would be required to address the remaining ones. For example, modifications to the database to store time zone offset information was required. Testing showed this method of changing the time zones caused unexpected downstream impacts to execution schedules and notification triggers, therefore further modifications were required.

It was also discovered that stored procedures and views which referenced deleted tables could not be migrated, and Azure SQL database did not allow their importation. Several iterations of the migration process were required to locate all the affected tables and remove them from Stored Procedures and Views before migration. Functionality built into SQL Server such as linked servers, FullText Search, and SQL CLR were not supported in Azure SQL and work around solutions were developed. At least ten different iterations of the migration process were required to discover these types of conflicts and develop strategies and solutions to overcome them.

Time-Based Schedulers

Experiments with Azure Scheduler were conducted to develop methods to configure and run scheduled jobs, however this service was retired by Microsoft leaving no alternative. Focus then shifted to develop a solution using Web Jobs, but it was discovered that its functionality was limited and required custom structures. Experiments to run time-based schedulers on the same application server as the web application resulted in performance issues within the web application. This finding resulted in the team shifting its focus to experiment with running jobs on isolated Azure ASPs, and it was found that functions could run properly.

**Technological Advancements** *(50 lines of 78 characters)*

By developing several code extensions and methods to overcome the limitations of the built-in Azure services, an automated procedure was developed that allowed for the complete migration of the SQL Server 2017 database to SQL Azure.

A method of time-based scheduling was developed by running scheduling functions on isolated Azure ASPs rather than on the same application server.