newtglobal

Aiming At Customer Delight

CASE STUDY

Application Migration and optimization on AWS

Newt Global Consulting LLC.
AMERICAS | INDIA

HQ Address:
1300 W Walnut Hill Ln | Suite # 230 | Irving TX 75038
Phone: +1 972 887 3165 | Fax: +1 214 260 6051

www.newtglobal.com/contactus

Introduction

Client is a Fortune 50 company and is one of the largest Telecom service providers in the US and offers multiple products and services to consumers and enterprises. Being a large enterprise, the IT foot-print is huge and there is a plethora of applications both COTS and internally developed. The vision of the Business and IT leadership teams is to migrate applications from Data centers to public cloud infra-structure in a phased manner. There is a big initiative to migrate legacy DBs to the public cloud as part of the DC revamp strategy. The client is looking at a few options for migrations that would be least disruptive and retain the performance characteristics. One such project is consolidation of multiple Oracle DBs into Aurora Postgres with limited rewriting of SPs and PL/SQL code.

.

Current System overview:

Client was running multiple instance of Databases and all are hosted ON-PREMISE. This has shot-up the license (covering Operating system license cost, DB server license cost) and maintenance cost. Added to this, servers could run into OS issued, as Solaris 10 (including support for oracle database) with end of Premium support date as of Sep 2018. They were also running the risk of one-point failure which could impact their business.

Client was also facing performance issue on these databases because of huge data and poor DB design.

Key Business/Systems requirements:

The Current application is the method of extending the clients Business network to the customer's premise using their own facility and /or third-party facility. The current application consists of cron jobs, ETL Batches, reports and several DB based batches and OLTP DB. Client wants to modernize the application by moving the workloads and leveraging AWS Elastic scaling technologies for the batch jobs, ETL and consolidating the DB in Postgres Aurora and potentially Cassandra on AWS in the future.

Solution Considerations

The Client has enumerated the following requirements for mandatory compliance while designing an approach to migration.

I. Automated DB scaling
II. Rewrite of ECOST application with micro-services architecture
III. Software Installation & Configuration (Tomcat, AWS Aurora Postgres, Jenkins 2.0, Java Springboot, Junit etc.)
IV. Automated Code Deployment
V. Scheduled Backup implementation
VI. DB AWS Migration:
    a. Consolidation to single machine in works by the existing team.
        i. ECOST DB also to be as one schema with rest of the apllication DB
        ii. Eliminate public Synonyms and Eliminate DB Links

iii. Evaluate if stored procedures need to be moved to spring boot or remain as Postgres Stored Procedures.

Batch Jobs: Complete Redesign

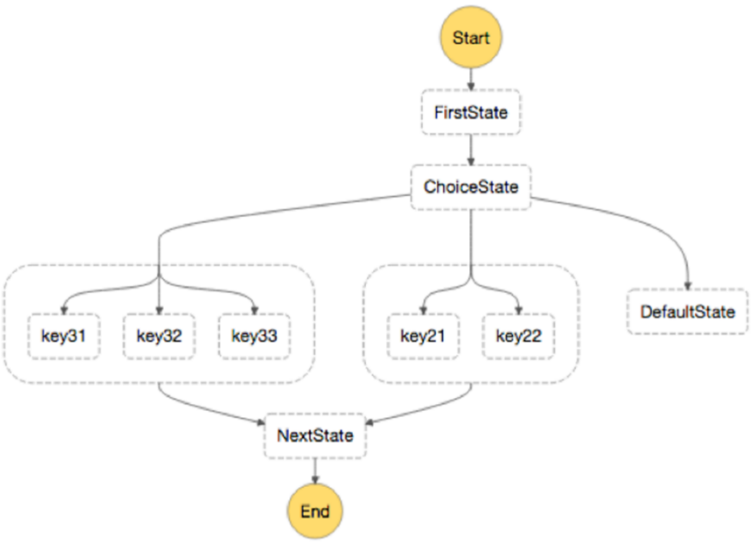
- Batch jobs driven by UI. Example use uploads csv of records to validate addresses.
- ETL Loads: Convert files (CSV, Logs, XLS, XLSX) and store to an RDBMS DB as of now and later to a no SQL DB such as Cassandra
- Maintenance Batches: Performs DB Maintenance: Purges and Deletes
- Reporting Batches: Generate Reports. Feed Reports to external systems
- Provide Visibility to all the batch Jobs.

Enhance the DB performance top 10 running SQL/PL SQL queries will be fine-tuned based on the outcome of AWR report.
Align the oracle tablespace to suite the need of AWS EC2 disk performance and their mount point's.
During the migration to cloud, strict enforcement of house-keeping of big audit trail tables will be attempted and ensure there is saving cost on the Storage disk.

Choice of Migration approach

Newt global has proposed to consolidate its all 7 databases into one and migrate it to cloud. This could help them to reduce spending on licensing. Since cloud model comes with best support and maintenance model, Client is now free from running its on premise infrastructure and spending on its maintenance. As cloud is also scalable, it is easy for the client to increase its infrastructure bandwidth as needed in future. And it is also paying for what is consumes, which becomes more cost effective for the client.
Since database is moved to cloud, its dependent applications and batch jobs around it are also migrated to cloud. This gives an opportunity to leverage other features available in cloud. Since services like S3 are on pay to use model this is also an cost effective option. AWS Data pipeline is

considered along with the associated components for ETL.



Proposed Technology stack:

Tools

- Tomcat
- AWS Aurora Postgres
- Cassandra
- JDK 1.8
- Jenkins 2.x
- Ansible 2.0.0
- Shell scripts

AWS specific Services

- AWS batch
- AWS Data Pipeline
- AWS CFT
- AWS Lambda
- AWS Aurora
- AWS EC2

Dev

- Java Spring boot
- Docker
- Angular JS

Implementation Phase:

Timelines

The client engagement was an end to and end DB migration to the AWS infra-structure with a scalable and highly available architecture.

Phase	Outcome	Elapsed time	Responsibility
Business Req. Study	Sign-off PRD	2 weeks	Newt/Client
Design	Sprint plan and deliverables	2 weeks	Newt
Development	Code base in AWS environment	10 weeks	Newt
Migration	Deployment pipeline	4 weeks	Newt/Client
UAT	Functionality & Performance	4 weeks	Newt/Client
Production cut-over	AWS configuration tweaking	4 weeks	Newt

Start of the project: March 20th, 2018
End of the project: August 25th, 2018.

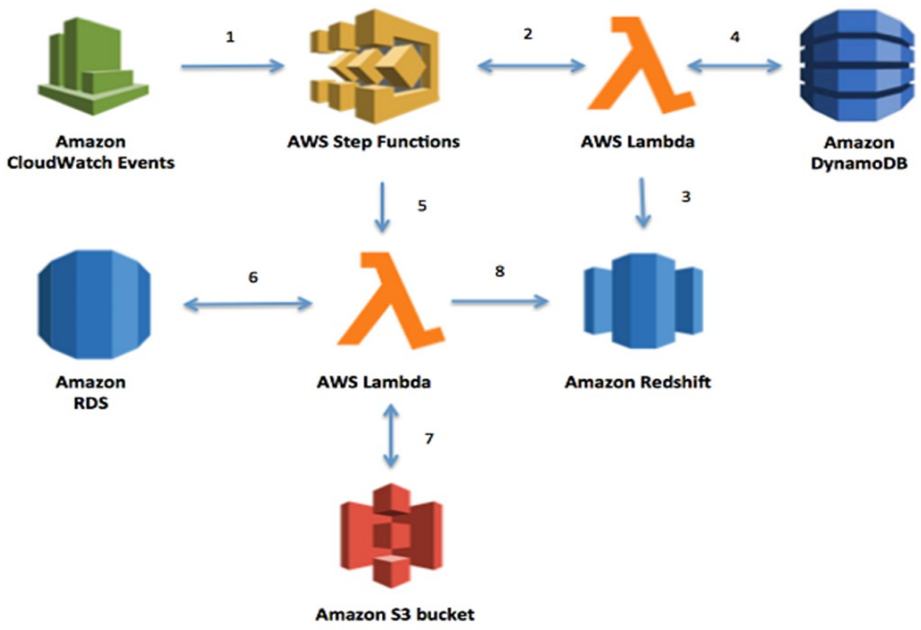
Team structure

The project team has 3 core Java developers, 1 Microservice Architect, 2 Java MS developers, I UX designer, 1 DB Architect, 2 DB Migration SMEs (Oracle/Postgres SQL), 4 testers and 2 AWS certified solution architects, 1 DBA and 1 AWS SysOps engineer for deployment and support.

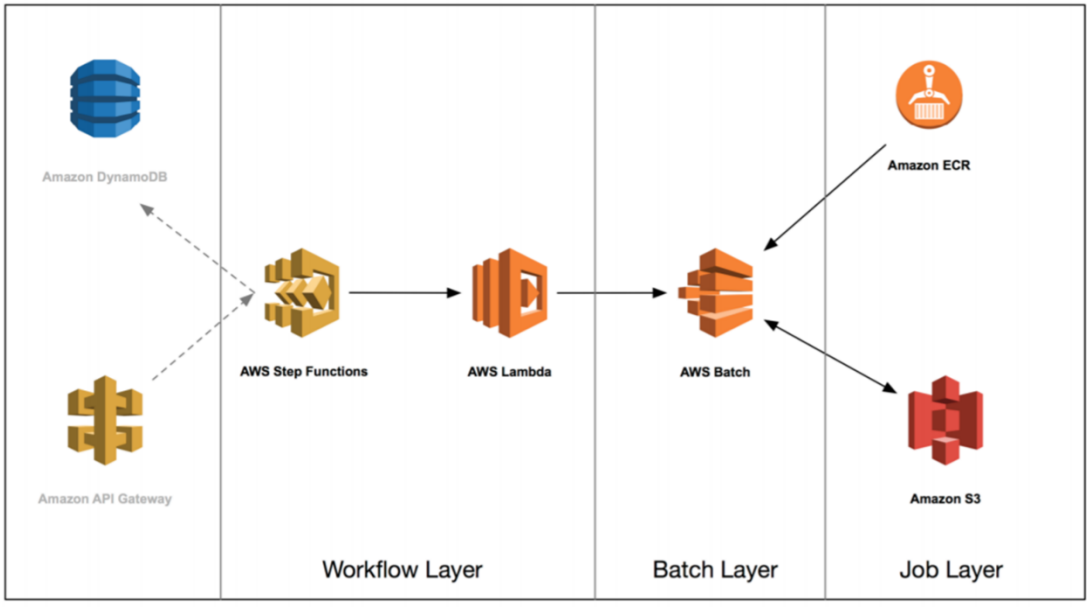
Implementation Phase

The deployment architecture is captured in the following sequence of events:

1. A rule in CloudWatch Events triggers the state machine execution on an automated schedule.
2. The state machine invokes the first Lambda function.
3. The Lambda function deletes all existing records in Amazon Redshift. Depending on the dataset, the Lambda function can create a new table in Amazon Redshift to hold the data.
4. The same Lambda function then retrieves Keys from a DynamoDB table. Keys represent specific business rules.
5. The state machine executes the second Lambda function using the Keys from DynamoDB.
6. The second Lambda function retrieves the referenced dataset from RDS. The records retrieved represent the entire dataset needed for a business process.
7. The second Lambda function executes in parallel for each Key retrieved from DynamoDB and stores the output in CSV format temporarily in S3.
8. Finally, the Lambda function uploads the data into Amazon Redshift



As part of the project the batch-file management using AWS Batch would be implemented as under:



AWS Service Used

AWS Step Functions, AWS Lambda, Amazon DynamoDB, Amazon RDS, Amazon Redshift, Amazon S3 and Amazon CloudWatch Events, EC2, S3 and ELB.

Third party tool/Products used

The application uses:
Jenkins/JIRA, Stash, Ajax/CSS, Angular JS, JDK, Tomcat, Eclipse, Java Spingboot, jUnit, JQuery, Oracle, Swagger, Maven, sl4j.

Security considerations and implementation

In line with the expectations of the application owner within the enterprise and also meeting the corporate security guidelines, the following best practices were implemented.

- ✓ VPC was adopted with NAT for enterprise access to EC2 instances
- ✓ All programming / API access to AWS was encrypted with TLS and user access was thru SSH.
- ✓ AMIs and deployment scripts were hardened as per business requirements
- ✓ Regular vulnerability scans were done
- ✓ IAM policies were baked with enterprise LDAP credentials.
- ✓ Data at rest was protected by using EBS and its native encryption

To be implemented before the closure of the project:

- ➢ Use of MFA with the clients' security token system
- ➢ Use automation to build workload images without requiring human intervention.
- ➢ A systematic workload re-provisioning strategy, since any workload could be compromised at any point.
- ➢ Use of CloudTrail for audit purposes

Business benefits of the migration:

- Provide scalability to batch process and Batch ETL
- Org visibility on batch jobs
- Leverage AWS and cloud scaling architectures