**Introduction:**

Neustar is world’s leading provider of surveillance, clearinghouse, and directory services to the global communications and Internet industry.

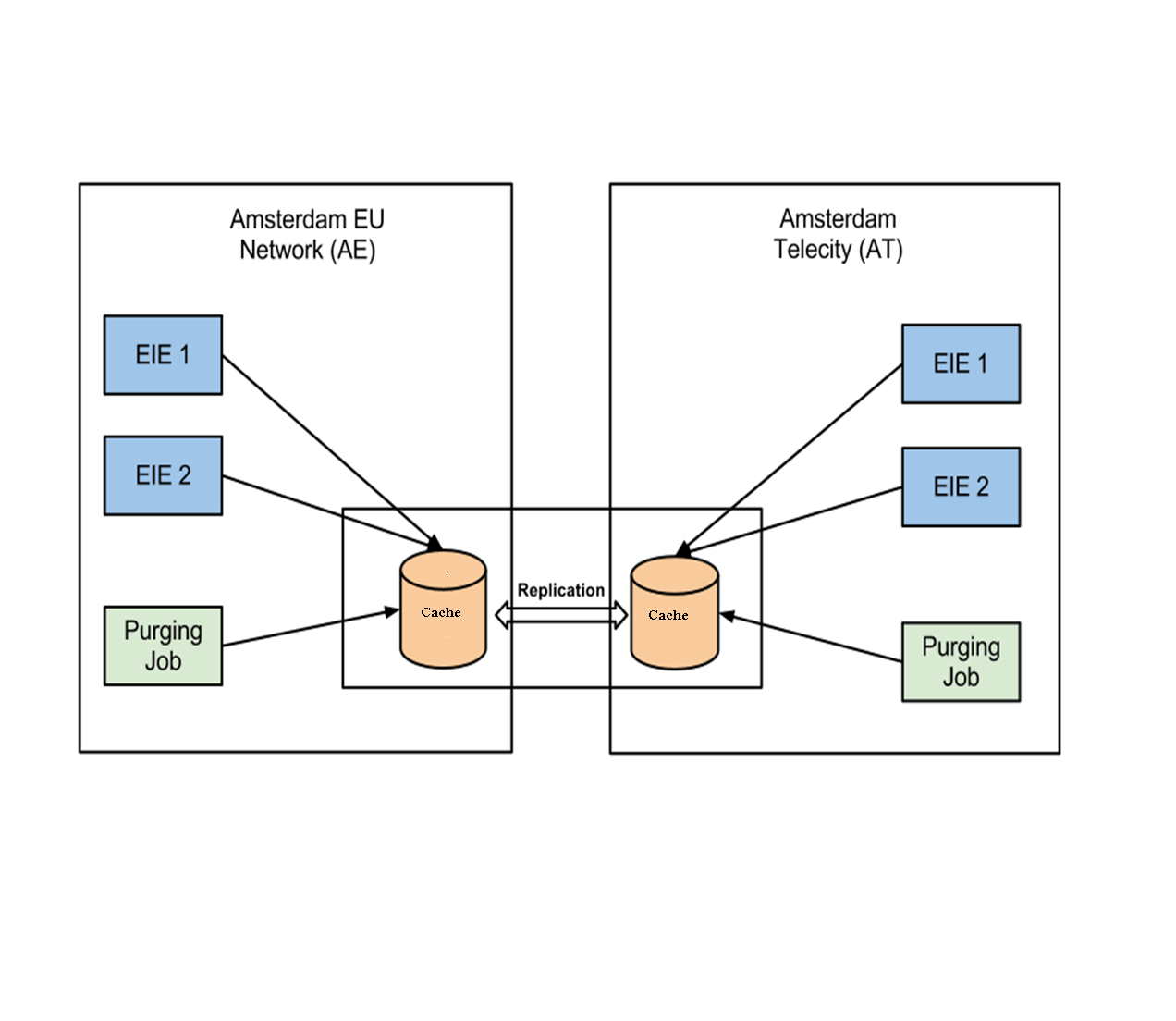
Neustar offers internal and external managed DNS services that play a key role in directing and managing traffic on the Internet, manages the authoritative directories for the [.us](http://en.wikipedia.org/wiki/.us) and [.biz](http://en.wikipedia.org/wiki/.biz) Internet domains, and acts as the worldwide registry gateway. Neustar manages a collection of these directories that maintain addresses to direct, prioritize and manage Internet traffic, and find and resolve Internet queries and top-level domains on behalf of its enterprise customers. Neustar serves as the provider of registry services and manages directories of similar resources, or addresses, that its customers use for access and connectivity.

**Existing System:**

External Interface Engine (EIE) is a component responsible for number lookup and query resolution for remote countries. EIE communicates with heterogeneous external interfaces for number lookup, this includes different kind of interfaces like ENUM (UDP), http, web service calls etc. These lookups primarily take Telephone number (TN) and country code (CC) in request and return porting information like MCC/MNC, IMSI, TN Type etc.

For cost optimizations and considering time to live (TTL) factor, EIE maintains a cache of Telephone numbers and before a number lookup against an external source, it dips in cache and if the TN is present in cache EIE responds from cache and does not query external interface.

Multiple instances of EIE are deployed on different sites. The instances on same site share the cache and across the sites. Currently the cache is implemented using Oracle table. The diagram shown below summarizes the deployment architecture -



# For further optimization and cost reduction, the cache across different sites need to synchronized, thus making this cache distributed in nature. Currently with the database table implementation, the synchronization is achieved by replication using custom tools.

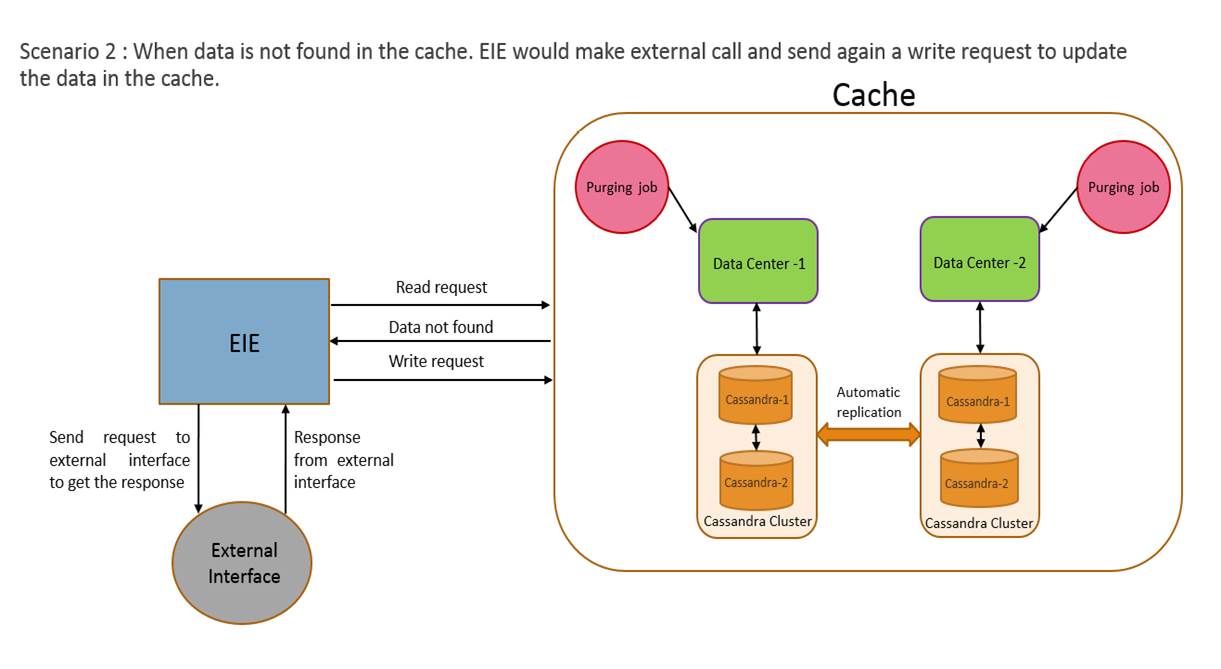
**Problem Statement:**

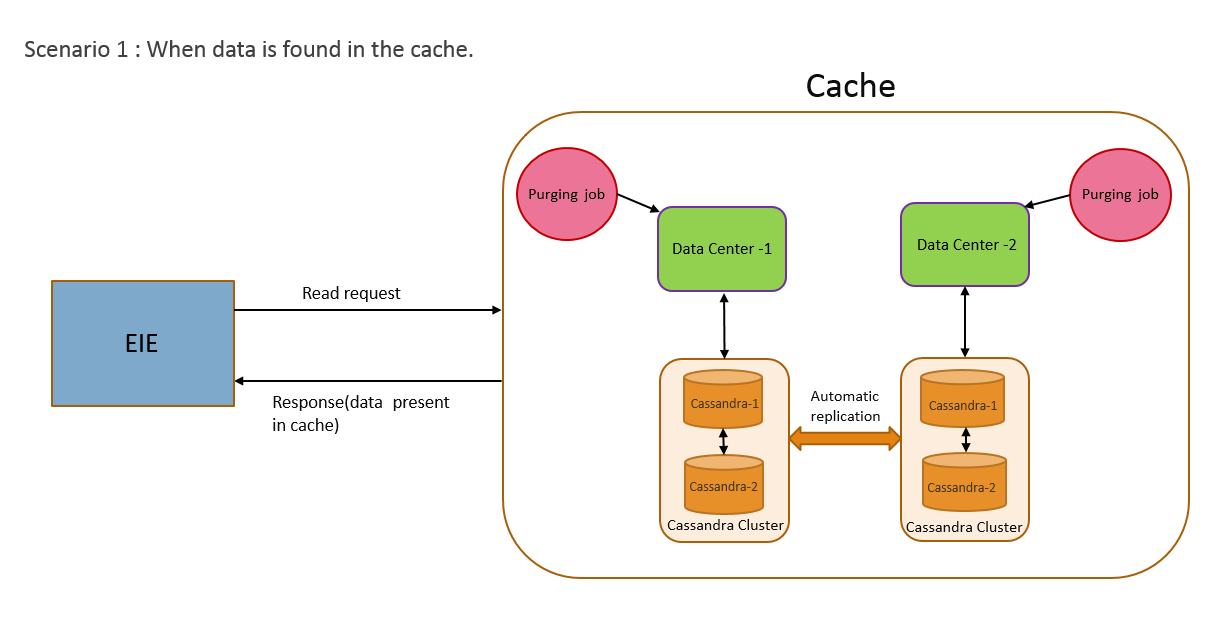
The existing cache system is implemented on Oracle RDBMS. In high concurrent access scenarios, the number of database connections becomes a bottleneck in case of database table based cache. The RDBMS implementation also suffers from high replication latency while synchronizing the data across multiple data centres.

**Proposed System:**

To overcome the above issues we are migrating the existing Oracle based cache implementation to NoSQL based cache implementation using Cassandra. Cassandra implementation facilitates continuous availability with built in synchronous replication support and linear performance gains while adding distributed data nodes.

**Usage Scenarios**

****For a given fetch request with a combination of (Country code, TN and Request Type) a cache lookup will be done. Every cache record will have a fixed TTL (time to live) associated. Post expiry of the TTL, the response will be discarded. The following are the two usage scenarios.



**Benchmark criteria**

* 2000+ queries per second
* Minimum possible latency for replication of nearly 50 million records