# Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** | **Reviewer** |
| 02/20/2015 | 1.0 | Initial draft | Rahul Kumar Sharma | Avinash Agrahari |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# Background

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

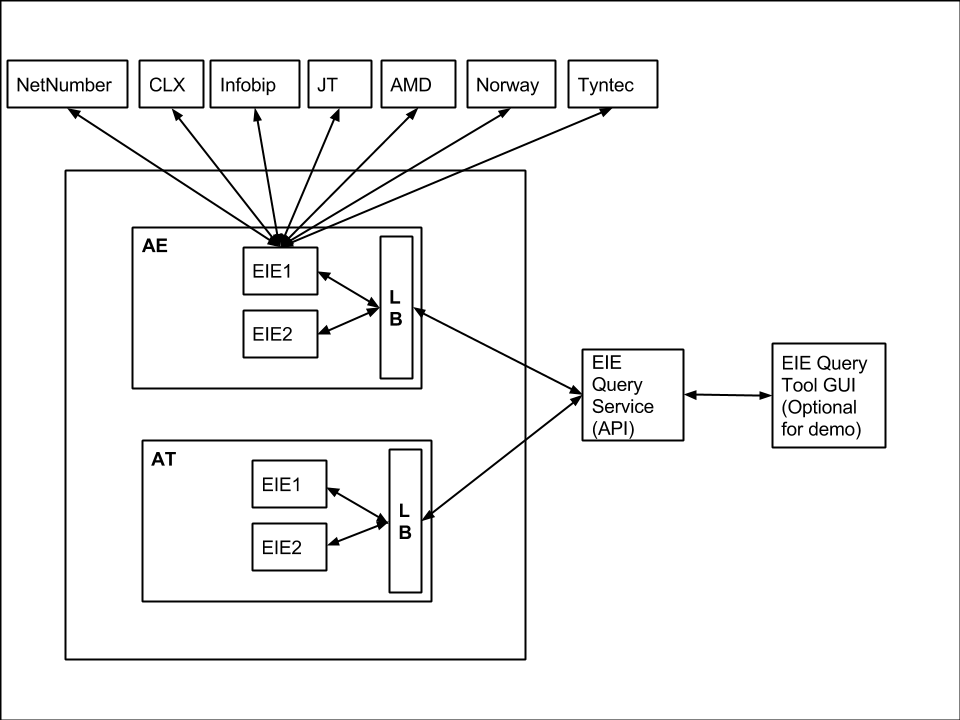
Neustar offers internal and external managed DNS services that play a key role in directing and managing traffic on the Internet, manages the authoritative directories for the [.us](http://en.wikipedia.org/wiki/.us) and [.biz](http://en.wikipedia.org/wiki/.biz) Internet domains, and acts as the worldwide 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.

# External Interface Engine (EIE)

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.

# EIE query service

The current interface exposed by EIE is based on a custom protocol based on socket communication using (TCP). This is a long time wish list item to support a lookup interface (REST based) that should have capability to provide single/bulk queries to EIE and collect the responses. The architecture diagram below shows the entire architecture of EIE, external interfaces and the new EIE query service API -



# Objective

The objective of this case study is to provide a REST based lookup service that should be able to support queries for single and bulk queries. The service should be able to accept the request parameters in JSON format and return the EIE response in JSON format. The service should also support optional values for some of the request parameters and should be able to communicate to multiple EIEs

# Requirements

1. Provide a REST based lookup service for EIE
2. The lookup service should use JSON format for request/response
3. The lookup service should support
   1. Single query
   2. Bulk query
   3. Optional values for some parameters (see next section Request/Response format for further details)
   4. Capturing the exact response provided by EIE
   5. Communicating with more than one EIEs

# Non Functional Requirements (NFRs)

The lookup service should be

1. Stateless (Each request to be served independently without relying/maintaining on any state information)
2. Scalable (Can be deployed on multiple sites to scale)
3. High performance (No hard/defined SLAs as such for response time but the processing time by API itself should be minimum. In case of request to multiple EIE optimizations can be made using threads, executors, concurrent tasks etc.)

# EIE request/response format

Following is the sample request format for EIE –

***CC=33 TN=649402336 TTL=0 NEG\_TTL=0 CUSTID=10000 SPN=52553 TIMEOUT=3000***

Details regarding request parameters –

|  |  |  |  |
| --- | --- | --- | --- |
| Request Parameter | Description | Type | Comments |
| CC | Country code | Numeric, max 3 digit |  |
| TN | Telephone number | Numeric, 5-15 digit |  |
| TTL | Time to live | Numeric | Optional, Default value 0 |
| Negative TTL | Time to live for negative caching | Numeric | Optional, Default value 0 |
| CUSTID | Customer ID | Numeric | Optional, Default value = 10000 |
| SPN | SPN | Numeric | Optional, Default value = 0 |
| TIMEOUT | Query timeout | Numeric | Optional, Default value = 3000 |

Following is the sample response format for EIE –

CC=33 TN=649402336 STATUS=0 CUSTID=10000 MCC=208 MNC=00 SPID=null IMSI=null HLR=null MSC=null TNTYPE=null STYPE=null SID=null VINT=CLX-ENUM

**Deliverables to be made**

|  |  |  |
| --- | --- | --- |
| S.No. | Deliverable | Priority |
| 1 | Requirement analysis document covering the understanding of system. Should include any assumptions, open questions and resolutions | P1 |
| 2 | Design document including design solution and considerations for above mentioned use cases | P1 |
| 3 | Complete EIE query service solution | P1 |
| 4 | Automated test suite and test execution results | P1 |
| 5 | Performance test plan | P2 |
| 6 | Test coverage, code review reports and java docs | P2 |
| 7 | Deployment& rollback plan for proposed service | P2 |
| 8 | GUI for demonstrating capabilities of EIE query tool (optional, browser plugins and/or extensions can be used) | P3 |

**Collaterals to be delivered**

1. Solution architecture including 4+1 View
2. The architecture should involve the industry best practices in requirement analysis, design, implementation, manual/automated testing, code review and deployment phases
3. Deployment plan torollout to different environments. Plan should include steps for incremental deployment and patch releases
4. Code base with API documentation, unit test cases, code review and test coverage reports
5. Test plan and test cases for the use cases implemented

# Suggested Technologies

1. Programming language–Java8
2. Integration framework –Spring/Drop wizard
3. Unit test framework – Junit/Test NG
4. Integration/scenario based testing framework – Cucumber, SoapUI, Frisbee etc.
5. Build tool – Maven/Gradle
6. Continuous Integration – Jenkins (Including test execution/code coverage/static analysis reports)
7. Packaging – tar/rpm
8. Deployment automation – Chef client/server
9. Performance test frame work - TBD

# Abbreviations and acronyms

|  |  |
| --- | --- |
| **Abbreviation/Acronym** | **Description** |
| EIE | External Interface Engine |
| TN | Telephone Number |
| CC | Country Code |

1. EIE is already developed and we have to develop REST Interface to consume the services produced by EIE? Or we have to develop EIE as well?
2. If we are going to develop EIE, then from where are will get the data to send a response? We need to populate some data base etc.?
3. Should be able to communicate to multiple EIEs, What does that mean? I am sorry but I could not understand the meaning of multiple EIEs, are we going to deploy EIE on multiple servers? In fact it would be really helpful if you explain the terms and acronyms used in Diagram.
4. Tools and technologies:
   1. I would suggest Spring, TestNG, Java8, gradle, jenkins

I suppose it’s like a Who is service for mobile numbers.