## **Document Attributes**

| Attribute | Value |
| --- | --- |
| Application ID /  Application Name | EDF - 23718 |
| Owner | Khan, Tofael |
| Owner Contact Information | Tk2412@att.com |
| Other Attribute |  |

## **Revision History**

The following table lists the revision history of this document:

| Author | Date | Version # | Revision Description |
| --- | --- | --- | --- |
| Tofael K/Akarsh V | 11/06/2018 | 0.01 | Initial draft HLD for 302755 |
| Tofael/Akarsh V | 11/20/2018 | 0.02 | Added “objectUId” in the “address” block. Changes are tracked with <302755-1> |
| Akarsh/Ravali | 12/26/2018 | 0.03 | Added Sub query as B3 for ACCESS\_CIRCUIT\_DETAILS table ETL load for the attributes of CUSTOMERLATA a.ka. Lata Code, ingressprofile, egressprofile, domainIndicataor a.k.a RegionFlag |
| Akarsh/Ravali | 12/27/2018 | 0.04 | 1. Added Acna,Ban,UbAccountId as new columns in tables CUST\_ACCT\_ORG\_DETAILS, CUSTOMER\_NETWORK\_DETAILS,  ACCESS\_CIRCUIT\_DETAILS  2. Added initial load logics for all the three tables for the fields -> Acna, Ban , UbAccountId.  3. Added NOT NULL conditions for  Cust\_ntwk\_ckt\_raw  Circuit\_format\_Raw  Cust\_ntwk\_ckt\_standard  in tables of CUST\_ACCT\_ORG\_DETAILS, CUSTOMER\_NETWORK\_DETAILS,  ACCESS\_CIRCUIT\_DETAILS  4. Added Note in the load of ACCESS\_CIRCUIT\_DETAILS.CosIngress  ACCESS\_CIRCUIT\_DETAILS.CosEgress  That asset will have either any of them, if both the field columns are null , then drop that record.  5. ACCESS\_CIRCUIT\_DETAILS.Lata\_Code updated as NOT NULL  6. ACCESS\_CIRCUIT\_DETAILS.Region Flag , added derivation logic to populating.  7. Added Not Null constraint for json mapping for City, address line 1 , country code . |
| Ravali/Akarsh | 12/31/2018 | 0.05 | Added derived logics for Nni\_vlan\_id  And MT Indicator logic. |
| Akarsh/Ravali | 1/03/2019 | 0.06 | Added filter to exclude UNIs where the UNI’s account is not registered under the BC customer. Change tagged with <v0.06-Update>  Provided note under initial load for first table , that need to pull the account orgs associated to a customer with check on whether any assets (EVC / uni) linked to those account orgs. |
| Akarsh/Ravali | 1/05/2018 | 0.07 | Corrected mapping for vlan\_bottom\_tag\_id , with derivation logic for table - **ACCESS\_CIRCUIT\_DETAILS**  Changes are tagged under <V0.07> |

## **Table of Contents**

[Overview 3](#_Toc458636495)

[Problem Statement 3](#_Toc458636496)

[Design Decisions 3](#_Toc458636497)

[Alternative Designs 6](#_Toc458636498)

[Assumptions/Risks 6](#_Toc458636499)

[Pre-Production Disaster Recovery Planning 6](#_Toc458636500)

[Other Plans and References 7](#_Toc458636501)

[Acceptance & Approvals.........................................................................................7](#_Toc458636502)

**Overview**

The High Level Design (HLD) describes how an application will implement the architectural concept and meet the requirements allocated to that application. The High Level Design describes the functionality the application will provide, the overall design for providing that functionality and meeting the nonfunctional requirements, and the rationale for choosing that design. The High Level Design also describes how the application will cooperate and interface with other applications to provide an integrated solution that achieves the architectural concept.

**Problem Statement**

This API is used to retrieve the GDB source system for an identified BC Organization which is present in GDB - which include asset, organization or Site etc. This information is required for BC to generate the SLA reports for showing the data on their screens to customer. The request for searching the data would be BC Organization present in GDB. Based on which the other hierarchical data should be identified. The assumption for this API is that the inventory information for the identified BC Org available and retrievable in the source system.

**Design Decisions**

It was decided to use DMaaP Message Q for processing and sending the required data in the form of JSON events once in a day after the data is loaded into GDB. Current understanding is that the CANOPI is the primary source for data being loaded into GDB and sent in response via DMaaP.

**Database Schema**

Table Name: CUST\_ACCT\_ORG\_DETAILS

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraint** |
| cust\_org\_id | varchar2 (25) | NOT NULL |
| acct\_org\_id | varchar2 (25) | NOT NULL |
| Acct\_org\_identifier\_name | Varchar2 (50) |  |
| Acna | Varchar2 (100) |  |
| Ban | Varchar2 (100) |  |
| ubAccountId | Varchar2 (100) |  |
| Cust\_org\_action\_indicator | Varchar2 (5) |  |
| Acct\_org\_action\_indicator | Varchar2 (5) |  |
| assoc\_contract\_number | varchar2 (100) |  |
| cust\_ntwk\_asset\_id | varchar2 (25) |  |
| Cust\_ntwk\_action\_indicator | Varchar2 (5) |  |
| Timestamp | timestamp (6) | NOT NULL |
| id\_change\_tracking | Number (20) | NOT NULL |

Table Name: CUSTOMER\_NETWORK\_DETAILS

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraint** |
| cust\_ntwk\_asset\_id | varchar2 (25) | NOT NULL (CK) |
| Access\_ckt\_asset\_id | varchar2 (100) | NOT NULL (CK) |
| Cust\_ntwk\_action\_indicator | Varchar2 (5) |  |
| Cust\_ntwk\_ckt\_raw | Varchar2 (25) | NOT NULL (CK) |
| Circuit\_format\_Raw | Varchar2 (25) | NOT NULL (CK) |
| Cust\_ntwk\_ckt\_standard | Varchar2 (25) | NOT NULL (CK) |
| Circuit\_Format\_Stnd | Varchar2 (25) |  |
| Access\_ckt\_action\_indicator | Varchar2 (5) |  |
| Acct\_org\_id | varchar2 (25) | NOT NULL |
| Acct\_org\_identifier\_name | Varchar2 (50) |  |
| Acna | Varchar2 (100) |  |
| Ban | Varchar2 (100) |  |
| UbAcountId | Varchar2 (100) |  |
| Acct\_org\_action\_indicator | Varchar2 (5) |  |
| Vpn\_id | varchar2 (25) |  |
| Vpn\_name | varchar2 (30) |  |
| Cust\_asset\_alias | Varchar2 (100) |  |
| Asset\_alias\_action\_indicator | Varchar2 (5) |  |
| Service\_type | Varchar2 (25) |  |
| Timestamp | timestamp (6) | NOT NULL |
| id\_change\_tracking | Number (20) | NOT NULL |

Table Name: ACCESS\_CIRCUIT\_DETAILS

|  |  |  |
| --- | --- | --- |
| **Column** | **Data Type** | **Constraint** |
| cust\_ntwk\_asset\_id | varchar2 (25) | NOT NULL (CK) |
| Access\_ckt\_asset\_id | varchar2 (100) | NOT NULL (CK) |
| Cust\_ntwk\_action\_indicator | Varchar2 (5) |  |
| access\_ckt\_raw | Varchar2 (25) | NOT NULL (CK) |
| Circuit\_format\_raw | Varchar2 (25) | NOT NULL (CK) |
| access\_ckt\_standard | Varchar2 (25) | NOT NULL (CK) |
| Circuit\_format\_stnd | Varchar2 (25) |  |
| Access\_ckt\_action\_indicator | Varchar2 (5) |  |
| Cust\_asset\_alias | Varchar2 (100) |  |
| Asset\_alias\_action\_indicator | Varchar2 (5) |  |
| Acct\_org\_id | varchar2 (25) | NOT NULL |
| Acct\_org\_action\_indicator | Varchar2 (5) | NOT NULL |
| Acct\_org\_identifier\_name | Varchar2 (50) |  |
| Acna | Varchar2(100) |  |
| Ban | Varchar2 (100) |  |
| UbAccountId | Varchar2 (100) |  |
| Assoc\_contract\_num | Varchar2 (100) |  |
| Contract\_action\_indicator | Varchar2 (5) |  |
| Service\_type | Varchar2 (25) |  |
| Physical\_port\_speed | Varchar2 (25) |  |
| Ingress\_cos\_percentage | Varchar2 (50) |  |
| Egress\_cos\_percentage | Varchar2 (50) |  |
| Site\_id | varchar2 (25) |  |
| Nte\_clli | Varchar2 (50) |  |
| Nte\_cos\_model | Varchar2 (30) |  |
| Top\_vlan\_tag\_id | Varchar2 (50) |  |
| Bottom\_vlan\_tag\_id | Varchar2 (50) |  |
| Nni\_vlan\_id | Varchar2 (20) |  |
| Lata\_code | Varchar2 (50) | NOT NULL |
| Region\_flag | Varchar2 (50) |  |
| Timestamp | timestamp (6) | NOT NULL |
| id\_change\_tracking | Number (20) | NOT NULL |

**End to End Flow for Initial Load**

This is the End to End Flow followed in this project

1. We need to identify all the BC Organization IDs loaded into GDB today via Organization identifier ‘CUSTOMER\_ORGANIZATION\_REPRESENTATION’.
2. Then we need to filter these BC Org IDs by finding the associated account organization IDs with identifiers ‘UB\_ACCOUNT\_ID’ or ‘ACNA\_BAN’.

Note : initial load for Customer table , needs to pull the account orgs associated to a customer with check on whether any assets (EVC / uni) linked to those account orgs.

1. For the identified account organization IDs, we should identify the associated asset.ID whose asset\_type is ‘CUSTOMER\_NETWORK’ (EVC) and corresponding elements for each of the asset. Also, we need to check for the service type – ‘ASE’ or ‘SDN-ETHERNET’ - associated for these assets.
2. For each of the identified assets (CUSTOMER\_NETWORK), we should identify the corresponding asset.IDs whose asset\_type is ‘ACCESS\_CIRCUIT’ (UNI) and corresponding elements including location details.
3. We need to identify the account org ID associated with each of the Asset (ACCESS\_CIRCUIT) as these assets can be associated with the same account org as asset (CUSTOMER\_NETWORK) or to a different account org.
4. A JSON event should be sent in the form of message Q.
5. The events can be sent at Organization(CUSTOMER) level, Asset(CUSTOMER\_NETWORK) level, Asset(ACCESS\_CIRCUIT) level in a sequence.
6. The JSON events could be shared across with the client few weeks prior to the deployment date so as to be ready for the delta changes coming in after deployment date.

Note:

1. The timing of when the jobs should be scheduled should be decided by dev team.

**End to End Flow for Delta Load**

This is the End to End Flow followed for delta load in this project is:

1. Three tables ‘CUST\_ACCT\_ORG\_DETAILS’, ‘CUSTOMER\_NETWORK\_DETAILS’ and ‘ACCESS\_CIRCUIT\_DETAILS’ are created.
2. The changes can be Insert/Create, Update or Delete that can be identified in GDB. The changes can be at identified at Organization(CUSTOMER), Organization(Account), Asset(CUSTOMER\_NETWORK), Asset(ACCESS\_CIRCUIT) or CustomerAssetAlias level.
3. The changes related to Organization(CUSTOMER) can be identified via the CreateEnterpriseOrganization or UpdateEnterpriseOrganization API through the change\_tracking.timestamp. The same should be updated in ‘CUST\_ACCT\_ORG\_DETAILS’ table. Also, the Organization(Account) to Asset(Customer\_Network) is identified via associations created during ETL load, based on which the table ‘CUST\_ACCT\_ORG\_DETAILS’ would be updated.
4. The changes related to Asset(Customer\_Network) or Asset(Access\_Circuit) can be identified via associations which happen during ETL load. In each of these cases, the changes should be captured in the tables CUSTOMER\_NETWORK\_DETAILS or ACCESS\_CIRCUIT\_DETAILS accordingly.
5. The changes related to CustomerAssetAlias is identified via the UpdateAlias API. The tables should be updated with the right customerAssetAlias value and corresponding action indicator as soon as the change is detected.
6. The JSON structure should be generated when each of these changes are identified in the staging tables based on the timestamp.

**Building the Response**

The logic to build the response is as follows.

There are different scenarios which could occur when we are checking the data at each level namely, Organization (CUSTOMER) level, Organization (ACCOUNT) level, Asset (CUSTOMER\_NETWORK) level, Asset (ACCESS\_CIRCUIT) level, customerAssetAlias.

We can identify the data at each level from GDB when the load is complete.

Identify the Organization (CUSTOMER) which is required for this project via the below mentioned associations.

ORGANIZATION (Customer) 🡨 (CONTRACTED\_BY) 🡨 ASSET 🡪 (IMPLEMENTED\_BY) 🡪 SERVICE (service\_type = ‘ASE’, ‘SDN-ETHERNET’)

For each of these Organization(Customer), we need to identify the different and load the following tables in GDB.

**Handling change tracking:**

Insert a record into the GDB.CHANGE\_TRACKING table using the id values corresponding to “changeUser” and “changeSystem” – as ‘GDB’;   
use the system date and time for CHANGE\_TIMESTAMP;  
keep the new GDB.CHANGE\_TRACKING.ID value (chgTrkId) for further reference

All the data identified above can be loaded into the new table CUST\_ACCT\_ORG\_DETAILS.

|  |  |  |
| --- | --- | --- |
| **GDB** | **GDB CUST\_ACCT\_ORG\_DETAILS** | **Processing Comments** |
| Organization.ID | cust\_org\_id | referencing ‘CUSTOMER\_ORGANIZATION\_REPRESENTATION’  Note: We should process only those records as per the association mentioned above. |
| Organization.ID | acct\_org\_id | referencing ‘SERVICE\_SPECIFIC\_CUSTOMER\_REPRESENTATION’ |
| Identifier\_Type.Type | acct\_org\_identifier\_name |  |
| Organisation\_Identifier\_Value.Value | ACNA | Organisation\_Identifier\_Type= ‘Acna\_ban’ |
| Organisation\_Identifier\_Value.Value | BAN | Organisation\_Identifier\_Type= ‘Acna\_ban’ |
| Organisation\_Identifier\_Value.Value | UbAccountId | Organisation\_Identifier\_Type= ‘UB\_AccountId’ |
|  | Cust\_org\_action\_Indicator | Valid Values:  C – Create or Insert  U – Update  D – Delete  N – No Change |
|  | Acct\_org\_action\_indicator | Valid Values:  C – Create or Insert  U – Update  D – Delete  N – No Change |
| facilitation\_Contract.contract\_number | assoc\_contract\_number | Organization(Account) -> (CREATED\_FOR) -> Facilitation\_Contract |
| asset.id | cust\_ntwk\_asset\_id |  |
|  | Cust\_ntwk\_action\_indicator | Valid Values:  C – Create or Insert  U – Update  D – Delete  N – No Change |
|  | Timestamp | Current timestamp |
|  | id\_change\_tracking | Identify the changed (inserted records and records having updated columns that are copied into the GDB) and process only changed data into the GDB.  Use METADATA.SOURCE\_PROCESS.name=’GDB’ for this process. |

For each of the identified Asset (CUSTOMER\_NETWORK), we should identify the corresponding Asset (ACCESS\_CIRCUIT) via the association

Asset (CUSTOMER\_NETWORK) -> (USED\_BY/NULL) -> Asset (NETWORK\_CONNECTION)

Asset (NETWORK\_CONNECTION) -> (PART\_OF/NULL) -> Asset (ACCESS\_CIRCUIT)

We can consider the assets which are loaded from CANOPI whose change\_system is ‘IDIS-TO-GDB’ only.

The elements and corresponding mappings which are required at each Asset (CUSTOMER\_NETWORK) and Asset (ACCESS\_CIRCUIT) are as follows:

1. **Data Mapping for Asset (CUSTOMER\_NETWORK) Elements:**

|  |  |  |
| --- | --- | --- |
| **Mapping** | **JSON Element** | **Processing Comments** |
|  | <evc> |  |
|  | <info> |  |
| cust\_ntwk\_asset\_id | assetUId |  |
| Cust\_ntwk\_action\_indicator | actionIdentifier |  |
|  | <identifier> |  |
| Vpn\_id | vpnId | referencing ‘CUSTOMER\_NETWORK’ |
| Vpn\_name | vpnName |  |
|  | </identifier> |  |
|  | <asset> |  |
|  | <details> |  |
|  | <circuitDetails> |  |
| Cust\_ntwk\_ckt\_raw  Or  Cust\_ntwk\_ckt\_standard | CircuitId | referencing CIRCUIT\_ID\_IDENTIFIER |
| Circuit\_format\_Raw  Or  Circuit\_Format\_Stnd | circuitIdFormat | if the circuitId is in standard format, then “STANDARD\_CIRCUIT\_ID\_IDENTIFIER”,  if the circuitId is in raw format, then “RAW\_CIRCUIT\_ID\_IDENTIFIER” |
| ALIAS\_VALUE.value | assetAlias |  |
|  | </circuitDetails> |  |
| Cust\_asset\_alias | serviceType | valid values: ‘ASE’, ‘SDN-ETHERNET’ |
|  | <account> |  |
| Acct\_org\_id | objectUId |  |
| Acct\_org\_action\_indicator | actionIdentifier |  |
|  | <identifier> |  |
| Acct\_org\_id | ubAccountId | If acct\_org\_identifier\_name = ‘UB\_ACCOUNT\_ID’ |
|  | <acnaBanId> |  |
| Acct\_org\_id | accessCareerNameAbbrevation | If acct\_org\_identifier\_name = ‘ACNA\_BAN’ |
| Acct\_org\_id | billingAccountNumber | If acct\_org\_identifier\_name = ‘ACNA\_BAN’ |
|  | </acnaBanId> |  |
|  | </identifier> |  |
|  | </account> |  |
|  | </details> |  |
|  | </asset> |  |
|  | </evc> |  |

|  |  |  |
| --- | --- | --- |
| **GDB** | **GDB CUSTOMER\_NETWORK\_DETAILS** | **Processing Comments** |
| Asset.id | cust\_ntwk\_asset\_id | referencing ‘CUSTOMER\_NETWORK’  ASSET 🡪 (IMPLEMENTED\_BY) 🡪 SERVICE (service\_type = ‘ASE’, ‘SDN-ETHERNET’) |
| Asset.id | Access\_ckt\_asset\_id | referencing ‘SERVICE\_SPECIFIC\_CUSTOMER\_REPRESENTATION’ |
|  | Cust\_ntwk\_action\_indicator | Valid Values:  C – Create or Insert  U – Update  D – Delete  N – No Change  These values will be set based on the addition or update or deletion happening during the load in to GDB. Dev can come up with a way to capture these changes. |
| Asset\_ext\_customer\_network.network\_id | Cust\_ntwk\_ckt\_raw | In Raw format |
|  | Circuit\_format\_raw | ‘RAW\_CIRCUIT\_ID\_IDENTIFIER’ |
| Asset\_ext\_customer\_network.network\_id | Cust\_ntwk\_ckt\_standard | In Standard format  converted to a “Standard US Domestic Circuit ID Format” Circuit ID value (see “Logic for converting a US Domestic circuit id value from ICORE into the standard US Domestic Circuit ID Format” in GCP-SA-HLD-for-GCP-GDB.For-ICORE-Load.docx) |
|  | Circuit\_Format\_Stnd | ‘STANDARD\_CIRCUIT\_ID\_IDENTIFIER’ |
|  | Access\_ckt\_action\_indicator | Valid Values:  C – Create or Insert  U – Update  D – Delete  N – No Change  These values will be set based on the addition or update or deletion happening during the load in to GDB. Dev can come up with a way to capture these changes. |
| Organization.id | Acct\_org\_id | Organization(Account) <- (CONTRACTED\_BY) <- Asset (CUSTOMER\_NETWORK) |
| Identifier\_type.type | Acct\_org\_identifier\_name |  |
|  | Acct\_org\_action\_indicator | Valid Values:  C – Create or Insert  U – Update  D – Delete  N – No Change  These values will be set based on the addition or update or deletion happening during the load in to GDB. Dev can come up with a way to capture these changes. |
| ~~Asset.id~~  GCP\_SERVICE.ServiceID | Vpn\_id | Referencing ‘CUSTOMER\_NETWORK’ |
| asset\_identifier\_value.value | Vpn\_name |  |
| Alias\_value.value | Cust\_asset\_alias | Check for the id\_alias\_value in alias\_association  Alias\_value.id = Alias\_association.id\_alias\_value for an id\_object which references the asset.id |
|  | Asset\_alias\_action\_indicator | Valid Values:  C – Create or Insert  U – Update  D – Delete  N – No Change  These values will be set based on the addition or update or deletion happening during the load in to GDB. Dev can come up with a way to capture these changes. |
| service\_type.type | Service\_type | Check for service via ASSET -> (IMPLEMENTED\_BY) -> SERVICE association |
| Organisation\_Identifier\_Value.Value | ACNA | Organisation\_Identifier\_Type= ‘Acna\_ban’ |
| Organisation\_Identifier\_Value.Value | BAN | Organisation\_Identifier\_Type= ‘Acna\_ban’ |
| Organisation\_Identifier\_Value.Value | UbAccountId | Organisation\_Identifier\_Type= ‘UB\_AccountId’ |
|  | id\_change\_tracking | Identify the changed (inserted records and records having updated columns that are copied into the GDB) and process only changed data into the GDB.  Use METADATA.SOURCE\_PROCESS.name=’GDB’ for this process. |

Database Traversals:

|  |  |
| --- | --- |
| Asset.ID | Id\_object\_what |
| Id\_object\_to | Organization.id |
| Organization.id | Organization\_identifier.id\_organization |
| Organization\_identifier.id\_identifier\_type | Identifier\_type.id |
| Asset.id | Id\_object\_what |
| Id\_object\_to | Service.id |
| Service.id\_service\_type | Service\_type.id |
| Asset.id (access\_circuit) | Id\_object\_what |
| Id\_object\_to | Asset.id (network\_connection) |
| Asset.id (network\_connection) | Id\_object\_what |
| Id\_object\_to | Asset.id (customer\_network) |
| Asset.id | Asset\_identifier.id\_asset |
| Asset\_identifier.id | Asset\_identifier\_value.id\_asset\_identifier |
| Asset\_identifier\_value.value | IDIS.GCP\_SERVICE.Name |

1. **Data Mapping for Asset (ACCESS\_CIRCUIT) Elements:**

|  |  |  |
| --- | --- | --- |
| **Mapping** | **JSON Element** | **Processing Comments** |
|  | <accessCircuit> |  |
|  | <info> |  |
| Access\_ckt\_asset\_id | assetUId |  |
| Access\_ckt\_action\_indicator | actionIdentifier |  |
|  | </info> |  |
|  | <asset> |  |
|  | <details> |  |
|  | <circuitDetails> |  |
| access\_ckt\_raw or access\_ckt\_standard | CircuitId | referencing CIRCUIT\_ID\_IDENTIFIER |
| Circuit\_format\_raw or circuit\_format\_stnd | circuitIdFormat |  |
| Cust\_asset\_alias | assetAlias |  |
|  | </circuitDetails> |  |
| Service\_type | serviceType | valid values: ‘ASE’, ‘SDN-ETHERNET’ |
| Physical\_port\_speed | physicalPortSpeed |  |
|  | <COSDetail> |  |
|  | <ingress> |  |
| Ingress\_cos\_percentage | cosPercentage |  |
|  | </ingress> |  |
|  | <egress> |  |
| Egress\_cos\_percentage | cosPercentage |  |
|  | </egress> |  |
|  | </COSDetail> |  |
| Nte\_clli | nteClli |  |
| Nte\_cos\_model | nteCOSModel |  |
| Top\_vlan\_tag\_id | topVlanTagId |  |
| Bottom\_vlan\_tag\_id | bottomVlanTagId |  |
| Nni\_vlan\_id | nniVlanId |  |
|  | </details> |  |
|  | <account> |  |
| Acct\_org\_id | objectUId |  |
| Acct\_org\_action\_indicator | actionIdentifier |  |
|  | <identifier> |  |
| ~~Acct\_org\_id~~  Organisation\_Identifier\_Value.Value | ubAccountId | If ~~acct\_org\_identifier\_name = ‘UB\_ACCOUNT\_ID’~~  Organisation\_Identifier\_Type= ‘UB\_Account\_Id’ |
|  | <acnaBanId> |  |
| ~~Acct\_org\_id~~  Organisation\_Identifier\_Value.Value | accessCareerNameAbbrevation | ~~If acct\_org\_identifier\_name = ‘ACNA\_BAN’~~  If acct\_org\_identifier\_name = ‘ACNA’ |
| ~~Acct\_org\_identifier\_name~~  Organisation\_Identifier\_Value.Value | billingAccountNumber | ~~If acct\_org\_identifier\_name = ‘ACNA\_BAN’~~  If acct\_org\_identifier\_name = ‘BAN’ |
|  | </acnaBanId> |  |
|  | </identifier> |  |
|  | <associatedContract> |  |
|  | <contract> |  |
| Assoc\_contract\_num | contractNumber |  |
| Contract\_action\_indicator | actionIdentifier |  |
|  | </contract> |  |
|  | </associatedContract> |  |
|  | </account> |  |
|  | <locations> |  |
|  | <address> |  |
| Note : For the below always given preference to unified address ( ID\_ADDRESS\_NOTATION\_UNIFIED is not null ). If this unified address is not found for a given asset then consider any one of the inventory address. | | |
| Address\_notation.id | objectUId | <302755-1> |
| address\_notation.country\_code | countryCodeISO3 | ISO3 Format only |
| address\_notation.country\_code | countryName | NOT NULL as in GDB. address\_notation.country\_code |
| address\_notation.state | Subdivision |  |
|  | <postalAddress> |  |
|  | <addressLine> |  |
| address\_notation.address\_line1 | addressLine1 | NOT NULL as in GDB.address\_notation.address\_line1 |
| address\_notation.address\_line2 | addressLine2 |  |
|  | addressLine3 | NULL |
|  | addressLine4 | NULL |
|  | addressLine5 | NULL |
|  | </addressLine> |  |
| address\_notation.city | City | NOT NULL as in GDB. address\_notation.city |
| address\_notation.postal\_code | postalCode |  |
| Lata\_code | lataCode |  |
| Region\_flag | regionFlag |  |
| address\_notation.latitude | Latitude |  |
| address\_notation.longitude | Longitude |  |
|  | </postalAddress> |  |
|  | </address> |  |
|  | </locations> |  |
|  | </asset> |  |
|  | </accessCircuit> |  |

|  |  |  |
| --- | --- | --- |
| **GDB** | **GDB ACCESS\_CIRCUIT\_DETAILS** | **Processing Comments** |
| Asset.id | cust\_ntwk\_asset\_id | referencing ‘CUSTOMER\_NETWORK’  ASSET 🡪 (IMPLEMENTED\_BY) 🡪 SERVICE (service\_type = ‘ASE’, ‘SDN-ETHERNET’) |
| Asset.id | Access\_ckt\_asset\_id | referencing ‘SERVICE\_SPECIFIC\_CUSTOMER\_REPRESENTATION’ |
|  | Cust\_ntwk\_action\_indicator | Valid Values:  C – Create or Insert  U – Update  D – Delete  N – No Change  These values will be set based on the addition or update or deletion happening during the load in to GDB. Dev can come up with a way to capture these changes. |
| Asset\_identifier\_value.value | access\_ckt\_raw | Referencing IDENTIFIER\_TYPE ‘CIRCUIT\_ID\_IDENTIFIER’ |
|  | Circuit\_format\_raw | ‘RAW\_CIRCUIT\_ID\_IDENTIFIER’ |
| Asset\_identifier\_value.value | access\_ckt\_standard | Referencing IDENTIFIER\_TYPE ‘STANDARD\_FORMAT\_CIRCUIT\_ID\_IDENTIFIER’ |
|  | Circuit\_format\_stnd | ‘STANDARD\_CIRCUIT\_ID\_IDENTIFIER’ |
|  | Access\_ckt\_action\_indicator | Valid Values:  C – Create or Insert  U – Update  D – Delete  N – No Change  These values will be set based on the addition or update or deletion happening during the load in to GDB. Dev can come up with a way to capture these changes. |
| Alias\_value.value | Cust\_asset\_alias | Check for the id\_alias\_value in alias\_association  Alias\_value.id = Alias\_association.id\_alias\_value for an id\_object which references the asset.id |
|  | Asset\_alias\_action\_indicator | Valid Values:  C – Create or Insert  U – Update  D – Delete  N – No Change  These values will be set based on the addition or update or deletion happening during the load in to GDB. Dev can come up with a way to capture these changes. |
| Organization.id | Acct\_org\_id | Organization(Customer)<-Organization(Account) <- (CONTRACTED\_BY) <- Asset |
|  | Acct\_org\_action\_indicator | Valid Values:  C – Create or Insert  U – Update  D – Delete  N – No Change  These values will be set based on the addition or update or deletion happening during the load in to GDB. Dev can come up with a way to capture these changes. |
| Identifier\_type.type | Acct\_org\_identifier\_name |  |
| Organisation\_Identifier\_Value.Value | ACNA | Organisation\_Identifier\_Type= ‘Acna’ |
| Organisation\_Identifier\_Value.Value | BAN | Organisation\_Identifier\_Type= ‘ban’ |
| Organisation\_Identifier\_Value.Value | UbAccountId | Organisation\_Identifier\_Type= ‘UB\_AccountId’ |
| Facilitation\_contract.contract\_number | Assoc\_contract\_num | Check for facilitation\_contract.id via Facilitation\_contract <- (CREATE\_FOR) <- Organization (Account) |
|  | Contract\_action\_indicator | Valid Values:  C – Create or Insert  U – Update  D – Delete  N – No Change  These values will be set based on the addition or update or deletion happening during the load in to GDB. Dev can come up with a way to capture these changes. |
| service\_type.type | Service\_type | Check for service via ASSET -> (IMPLEMENTED\_BY) -> SERVICE association |
| Asset\_Ext\_Access\_Circuit.Physical\_Port\_Speed | Physical\_port\_speed |  |
| IDIS Service. COSIngressProfile | Ingress\_cos\_percentage | Note : Asset will have either any of Ingress or Egress . For an asset if both Ingress and Egress are found null, drop that asset . |
| IDIS Service. COSEgressProfile | Egress\_cos\_percentage | Note : Asset will have either any of Ingress or Egress . For an asset if both Ingress and Egress are found null, drop that asset . |
| Site.id | Site\_id | Check for site id via ASSET -> (PART\_OF) -> SITE |
| IDIS.Gcp\_Location.Name | Nte\_clli | for Node\_Type ‘NTE’ |
|  | Nte\_cos\_model | Derivation logic given below |
|  | Top\_vlan\_tag\_id | <empty> |
| ~~Gcp\_number.num~~  <V0.07>  GCP\_number.name | Bottom\_vlan\_tag\_id | GCP\_number.name  <V0.07>  The Customer VLAN id (CVLanId) will be used as Bottom VLAN Tag ID  SELECT DISTINCT  num.name as CEVlanId  FROM  GCP\_CIRCUITCIRCUIT eth\_vlan,  GCP\_CIRCUIT vlan,  GCP\_CIRCUIT eth,  GCP\_NUMBEROBJECTS no,  GCP\_NUMBER num  WHERE (1=1)  And eth.CircuitType like 'ETHERNET CONNECTION%'  and eth\_vlan.circuitid = eth.circuitid  and eth\_vlan.usescircuitid = vlan.circuitid  and vlan.name like 'CVLAN%'  and vlan.circuittype = 'VLAN SEGMENT'  and no.RelatedObjecttype = 'CIRCUIT'  and no.numberid = num.numberid  and num.NumberType = 'VLAN'  and no.relatedobjectid = vlan.circuitid  and eth.circuitid = < TRIM(SERVICEOBJECTS.RELATEDOBJECTID) from the feed query from page no : 25 > |
| Gcp\_number.num | Nni\_vlan\_id | Database traversal : Map A5.CircuitId ( Its is derived logic refer to Nni\_vlan\_id ) with ACESS\_CiRCUIT \_DETAILS.Access\_ckt\_asset\_id  Note – must be the VLANID of the NNI associated with A/Z-End NTE UNI. If MT indicator is”N”, VLAND ID is default to 4092 |
| IDIS.Cloud.Name | Lata\_code |  |
| IDIS.Gcp\_service.domainindicator | Region\_flag | Refer to derivation logic for Region Flag in page no : 16. |
|  | Timestamp | Current timestamp |
|  | ID\_Change\_Tracking | Identify the changed (inserted records and records having updated columns that are copied into the GDB) and process only changed data into the GDB.  Use METADATA.SOURCE\_PROCESS.name=’GDB’ for this process. |

Database Traversals:

|  |  |
| --- | --- |
| Asset.ID | Id\_object\_what |
| Id\_object\_to | Organization.id |
| Organization.id | Organization\_identifier.id\_organization |
| Organization\_identifier.id\_identifier\_type | Identifier\_type.id |
| Asset.id | Id\_object\_what |
| Id\_object\_to | Service.id |
| Service.id\_service\_type | Service\_type.id |
| Asset.id (access\_circuit) | Id\_object\_what |
| Id\_object\_to | Asset.id (network\_connection) |
| Asset.id (network\_connection) | Id\_object\_what |
| Id\_object\_to | Asset.id (customer\_network) |
| Asset.id | Asset\_identifier.id\_asset |
| Asset\_identifier.id | Asset\_identifier\_value.id\_asset\_identifier |
| Asset.id | Alias\_association.id\_object |
| Organization.id | Id\_object\_what |
| Id\_object\_to | Facilitation\_contract.id |
| Asset.id | Asset\_ext\_access\_circuit.id\_asset |
| Asset\_ext\_access\_circuit.circuit\_id\_value | Gcp\_circuit.name |
| GCP\_SERVICEOBJECTS.serviceid | GCP\_SERVICE[evc].serviceid |
| GCP\_SERVICEOBJECTS.relatedobjectid | GCP\_SERVICE[leg\_evc].serviceid |
| GCP\_SERVICEOBJECTS[leg\_evc].serviceid | GCP\_SERVICE[leg\_evc].serviceid |
| GCP\_SERVICEOBJECTS[leg\_evc].relatedobjectid | GCP\_CIRCUIT.circuitid |
| GCP\_LOCATION.locationid | GCP\_CIRCUIT.endlocation |
| GCP\_CIRCUITCIRCUIT.usescircuitid | GCP\_NUMBEROBJECTS.relatedobjectid |
| GCP\_NUMBER.numberid | GCP\_NUMBEROBJECTS.numberid |
| Asset.id | Id\_object\_what |
| Id\_object\_to | Site.id |

<v0.06-Update>

Ensure that the UNI (Asset (ACCESS\_CIRCUIT)) belongs to the same customer as the EVC (Asset (CUSTOMER\_NETWORK)) by checking:

Asset (ACCESS\_CIRCUIT) 🡪 (CONTRACTED\_BY/NULL) 🡪 ORGANIZATION (Account) 🡪 (ROLLS\_UP\_TO/…) 🡪 ORGANIZATION (Customer)

</v0.06-Update>

**Derivation logic for Nni\_vlan\_id :**

This is the VLAN associated to the NNI, it’s used to identify NNI-1 and NNI-2 :

NM-VLAN (contains 4092) is 1st NNI, SLA2-VLAN (contains 4091) is 2nd NNI

Sample values:

4092

4091

A: If MT indicator is”N”, VLAND ID is default to 4092  
B: If MT indicator is <> "N", get the VLANID with following SQL:

For uniCicrcuitID use:

- Circuit2.CircuitID from Step#5 (ASE)

- eb.circuitid from Step#50 (FBS)

Select distinct

c1.Name AS VlanID

From gcp\_Circuit c1,

gcp\_circuit uniCkt,

gcp\_circuit clfi,

gcp\_SERVICE unisvc,

gcp\_SERVICEOBJECTS svcobj,

gcp\_circuitcircuit circir

Where (1=1)

and C1.Circuittype ='VLAN SEGMENT'

and (C1.Name like '%4092' or C1.Name like '%4091')

and unisvc.servicetype='UNI SERVICE'

and unisvc.serviceid=svcobj.serviceid

and svcobj.relatedobjecttype='CIRCUIT'

and svcobj.relatedobjectid=unickt.circuitid

and clfi.circuitid = unisvc.relatedCNL

and circir.circuitid = c1.circuitid

and circir.usescircuitid = clfi.circuitid

and uniCkt.circuitid = A5.CircuitID

If C1.Name contains NMV4091 : VlanId = 4091

If C1.Name contains NMV4092 : VlanId = 4092

Sample C1.Name:

JU101/GE1N/ORLDFLPH51W/ORLHFLSGH01\_NMV4091 🡺 VLANID = 4091  
JU102/GE1N/ORLDFLMA51W/ORLHFLSGH01\_NMV4092 🡺 VLANID = 4092  
  
If more than one record found : use the first one  
if no record was found: use 4092 as Default VLANID

**For MT Indicator logic :**

For NodeID use:

- the StartNode or EndNode Circuit2.NTENode (Node3.NodeID) from step 5 (ASE)

- the emtNode.nodeid from Step#50 (FBS)

Retrieve MT Indicator flag as below :

Select multiNNI~~, Oversubscribed~~ from gcp\_node\_ext where nodeid = <Node3.nodeid from Step 5>

Note: Set MTIndicator ~~and NTE Oversubscribed Indicator~~ to ‘N’ if no record was found – use ‘N’ as default.

**Derivation logic for RegionFlag :**

|  |  |
| --- | --- |
| **Canopi database field** | **Value region indicator** |
| Service.DomainIndicator = ‘OOR’ | O |
| 5th/6th char in Location.name (A-End IPAG CLLI) in  NC, SC, GA, KY, TN, LA, AL, MS, FL | B |
| 5th/6th char in Location.name (A-End IPAG CLLI) in  TX, OK, KS, MO, AR, CA, NV, WI, IL, IN, OH, MI, CT | S |
| Service.DomainIndicator **<>** ‘OOR’ AND  5th/6th char in location.name (A-End IPAG CLLI) NOT in  TX, OK, KS, MO, AR, CA, NV, WI, IL, IN, OH, MI, CT,  NC, SC, GA, KY, TN, LA, AL, MS, FL | IPAG Location will NOT be added to the output file; record will be skipped |

Derivation Logic for nteCOSModel:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | IDIS  Service.CosModel | IDIS  Service.CosCategory | IDIS-Service.CosSubCategory | CoS Model  (derived) |
| 1 | Cos4 | SGOS | N/A (no need to check) | 4GOS |
| 2 | Cos6 | SGOS | N/A (no need to check) | 6GOS |
| 3 | Cos6 | PPCOS | SES | PPCOS-SES |
| 4 | Cos6 | PPCOS | MES | PPCOS-MES |
| 5 | For any combination of values other than the ones listed above in row 1, 2, 3, and 4 |  |  | Skip the complete record; don’t create a GPR record in the feed file |

A sample query to identify the data for nte and lata. Dev can work and prepare a better code for performance improvement

~~SELECT DISTINCT~~

~~EVC.SERVICEID AS SERVICEID,~~

~~LOCB8.NAME AS NTECLLI,~~

~~CLOUD.NAME AS CUSTOMERLATA,~~

~~LEG.COSMODEL,~~

~~LEG.COSINGRESSPROFILE,~~

~~LEG.COSEGRESSPROFILE,~~

~~UNI.PRODUCTTYPE AS UNI\_PRODUCTTYPE,~~

~~UNI.DOMAININDICATOR ,~~

~~LOCA8.NAME--,~~

~~/\*EVC.PRODUCTTYPE AS EVC\_PRODUCTTYPE ,~~

~~CASE~~

~~WHEN UNI.PRODUCTTYPE ='SDN-ETHERNET' THEN UNI\_SUBSCRIBER.CUSTOMERNUMBER~~

~~ELSE ''~~

~~END AS UBID\*/~~

~~FROM~~

~~GCP\_SERVICE EVC,~~

~~GCP\_SERVICE LEG,~~

~~GCP\_SERVICE UNI,~~

~~GCP\_SERVICE\_EXT EXT,~~

~~GCP\_SERVICEOBJECTS EVC\_LEG,~~

~~GCP\_SERVICEOBJECTS LEG\_UNI,~~

~~GCP\_SUBSCRIBER SUBSCRIBER,~~

~~GCP\_SUBSCRIBER UNI\_SUBSCRIBER,~~

~~GCP\_SERVICEOBJECTS SERVICEOBJECTS,~~

~~(~~

~~SELECT~~

~~NODE1.NODEID AS CIRCUIT1\_MDFNODE,~~

~~NODE1.NODETYPE,~~

~~CIRCUIT1.CIRCUITID,~~

~~CIRCUIT1.STARTLOCATION,~~

~~CIRCUIT1.ENDLOCATION~~

~~FROM~~

~~GCP\_NODE NODE1,~~

~~GCP\_CIRCUIT CIRCUIT1~~

~~WHERE NODE1.NODEID=CIRCUIT1.STARTNODE~~

~~AND NODE1.NODETYPE LIKE 'MDF%'~~

~~AND CIRCUIT1.CIRCUITTYPE LIKE 'ETHERNET CONNECTION%'~~

~~UNION~~

~~SELECT~~

~~NODE1.NODEID AS CIRCUIT1\_MDFNODE,~~

~~NODE1.NODETYPE,~~

~~CIRCUIT1.CIRCUITID,~~

~~CIRCUIT1.STARTLOCATION,~~

~~CIRCUIT1.ENDLOCATION~~

~~FROM GCP\_NODE NODE1,~~

~~GCP\_CIRCUIT CIRCUIT1~~

~~WHERE~~

~~NODE1.NODEID=CIRCUIT1.ENDNODE~~

~~AND NODE1.NODETYPE LIKE 'MDF%'~~

~~AND CIRCUIT1.CIRCUITTYPE LIKE 'ETHERNET CONNECTION%'~~

~~) A3,~~

~~(~~

~~SELECT~~

~~NODE3.NODEID AS CIRCUIT2\_NTENODE,~~

~~NODE3.NODETYPE,~~

~~CIRCUIT2.CIRCUITID,~~

~~NODE3.DEVICESUBROLE,~~

~~CIRCUIT2.STARTLOCATION AS CIRCUIT2\_NTELOCATION,~~

~~CIRCUIT2.ENDNODE AS CKTND,~~

~~CIRCUIT2.NAME,~~

~~CIRCUIT2.AAFDAROLE~~

~~FROM~~

~~GCP\_NODE NODE3,~~

~~GCP\_CIRCUIT CIRCUIT2~~

~~WHERE~~

~~NODE3.NODEID=CIRCUIT2.STARTNODE~~

~~AND (~~

~~upper(node3.NodeType) in ('ADTRAN 800 SERIES','CISCO 900 SERIES') or~~

~~(upper(node3.NodeType) in ('CIENA 3900 SERIES','CIENA 8700 SERIES') and upper(node3.deviceRole) = 'NTE')~~

~~Or~~

~~(upper(node3.NodeType) = 'CIENA EMUX' and upper(node3.devicesubrole) = 'EMT')~~

~~)~~

~~AND CIRCUIT2.CIRCUITTYPE IN ('TIRKS BEARER','UNROUTED ETHERNET BEARER','ETHERNET BEARER')~~

~~UNION~~

~~SELECT~~

~~NODE3.NODEID AS CIRCUIT2\_NTENODE,~~

~~NODE3.NODETYPE,~~

~~CIRCUIT2.CIRCUITID,~~

~~NODE3.DEVICESUBROLE,~~

~~CIRCUIT2.ENDLOCATION AS CIRCUIT2\_NTELOCATION ,~~

~~CIRCUIT2.STARTNODE AS CKTND,~~

~~CIRCUIT2.NAME,~~

~~CIRCUIT2.AAFDAROLE~~

~~FROM~~

~~GCP\_NODE NODE3,~~

~~GCP\_CIRCUIT CIRCUIT2~~

~~WHERE~~

~~NODE3.NODEID=CIRCUIT2.ENDNODE~~

~~AND (~~

~~upper(node3.NodeType) in ('ADTRAN 800 SERIES','CISCO 900 SERIES') or~~

~~(upper(node3.NodeType) in ('CIENA 3900 SERIES','CIENA 8700 SERIES') and upper(node3.deviceRole) = 'NTE')~~

~~Or~~

~~(upper(node3.NodeType) = 'CIENA EMUX' and upper(node3.devicesubrole) = 'EMT')~~

~~)~~

~~AND CIRCUIT2.CIRCUITTYPE IN ('TIRKS BEARER','UNROUTED ETHERNET BEARER','ETHERNET BEARER')~~

~~) A5 ,~~

~~GCP\_NUMBEROBJECTS NUMOBJS ,~~

~~GCP\_NUMBER NUMBER1 ,~~

~~GCP\_LOCATION LOCA8,~~

~~GCP\_LOCATION LOCB8,~~

~~GCP\_CLOUDOBJECTS CLOUDOBJECTS,~~

~~GCP\_CLOUD CLOUD~~

~~WHERE~~

~~SERVICEOBJECTS.SERVICEID = LEG.SERVICEID~~

~~AND SERVICEOBJECTS.RELATEDOBJECTTYPE = 'CIRCUIT'~~

~~AND A3.CIRCUITID = SERVICEOBJECTS.RELATEDOBJECTID~~

~~--AND B3.CIRCUITID = SERVICEOBJECTS.RELATEDOBJECTID~~

~~AND A5.CKTND = A3.CIRCUIT1\_MDFNODE~~

~~AND NUMOBJS.RELATEDOBJECTTYPE = 'NODE'~~

~~AND NUMOBJS.RELATEDOBJECTID = A5.CIRCUIT2\_NTENODE~~

~~AND NUMBER1.NUMBERID = NUMOBJS.NUMBERID~~

~~AND NUMBER1.NUMBERTYPE in ('ADMIN ETHERNET IP ADDRESS','IPV4 TWAMP IP ADDRESS')~~

~~AND NUMBER1.NAME LIKE '%.%'~~

~~--AND LOCA8.LOCATIONID = B3.CIRCUIT1\_IPAGLOCATION~~

~~AND LOCB8.LOCATIONID = A5.CIRCUIT2\_NTELOCATION~~

~~--AND CLOUDOBJECTS.RELATEDOBJECTID = B3.CIRCUIT1\_IPAGNODE~~

~~AND CLOUDOBJECTS.RELATEDOBJECTTYPE = 'NODE'~~

~~AND CLOUDOBJECTS.CLOUDID = CLOUD.CLOUDID~~

~~AND CLOUD.TYPE = 'LATA'~~

~~AND EVC.SUBSCRIBERID=SUBSCRIBER.SUBSCRIBERID~~

~~--AND SVC\_EXT.SERVICEID = EVC.SERVICEID~~

~~AND UNI.SERVICETYPE='UNI SERVICE'~~

~~AND UNI.PRODUCTTYPE IN ('OEM-AC','SDN-ETHERNET')~~

~~AND EVC.SERVICETYPE='EVC SERVICE'~~

~~AND LEG.SERVICETYPE='LEG EVC SERVICE'~~

~~AND EVC.SERVICEID=EVC\_LEG.SERVICEID~~

~~AND EVC\_LEG.RELATEDOBJECTID= LEG.SERVICEID~~

~~AND EVC\_LEG.RELATEDOBJECTTYPE='SERVICE'~~

~~AND LEG.SERVICEID=LEG\_UNI.SERVICEID~~

~~AND LEG\_UNI.RELATEDOBJECTID= UNI.SERVICEID~~

~~AND LEG\_UNI.RELATEDOBJECTTYPE='SERVICE'~~

~~AND A5.NAME = UNI.NAME~~

~~AND LEG."In-Effect\_Indicator" = 'Y'~~

~~AND UNI.SUBSCRIBERID = UNI\_SUBSCRIBER.SUBSCRIBERID (+)~~

~~AND EVC.SERVICEID = EXT.SERVICEID (+) ;~~

SELECT DISTINCT

EVC.SERVICEID AS SERVICEID,

LOCB8.NAME AS NTECLLI,

CLOUD.NAME AS CUSTOMERLATA,

LEG.COSMODEL,

LEG.COSINGRESSPROFILE,

LEG.COSEGRESSPROFILE,

UNI.PRODUCTTYPE AS UNI\_PRODUCTTYPE,

UNI.DOMAININDICATOR ,

LOCA8.NAME,

EVC.PRODUCTTYPE AS EVC\_PRODUCTTYPE ,

CASE

WHEN UNI.PRODUCTTYPE ='SDN-ETHERNET' THEN UNI\_SUBSCRIBER.CUSTOMERNUMBER

ELSE ''

END AS UBID,

A5.circuitID

FROM

idis.GCP\_SERVICE EVC,

idis.GCP\_SERVICE LEG,

idis.GCP\_SERVICE UNI,

idis.GCP\_SERVICE\_EXT EXT, --added as a part of 272078d

idis.GCP\_SERVICEOBJECTS EVC\_LEG,

idis.GCP\_SERVICEOBJECTS LEG\_UNI,

idis.GCP\_SUBSCRIBER SUBSCRIBER,

idis.GCP\_SUBSCRIBER UNI\_SUBSCRIBER, -- added as a part of 272078d

idis.GCP\_SERVICEOBJECTS SERVICEOBJECTS,

(

SELECT

NODE1.NODEID AS CIRCUIT1\_MDFNODE,

NODE1.NODETYPE,

CIRCUIT1.CIRCUITID,

CIRCUIT1.STARTLOCATION,

CIRCUIT1.ENDLOCATION

FROM

idis.GCP\_NODE NODE1,

idis.GCP\_CIRCUIT CIRCUIT1

WHERE NODE1.NODEID=CIRCUIT1.STARTNODE

AND NODE1.NODETYPE LIKE 'MDF%'

AND CIRCUIT1.CIRCUITTYPE LIKE 'ETHERNET CONNECTION%'

UNION

SELECT

NODE1.NODEID AS CIRCUIT1\_MDFNODE,

NODE1.NODETYPE,

CIRCUIT1.CIRCUITID,

CIRCUIT1.STARTLOCATION,

CIRCUIT1.ENDLOCATION

FROM idis.GCP\_NODE NODE1,

idis.GCP\_CIRCUIT CIRCUIT1

WHERE

NODE1.NODEID=CIRCUIT1.ENDNODE

AND NODE1.NODETYPE LIKE 'MDF%'

AND CIRCUIT1.CIRCUITTYPE LIKE 'ETHERNET CONNECTION%'

) A3 ,

(

SELECT

NODE2.NODEID AS CIRCUIT1\_IPAGNODE,

CIRCUIT1.CIRCUITID,

CIRCUIT1.STARTLOCATION AS CIRCUIT1\_IPAGLOCATION,

CIRCUIT1.STARTNODE AS CIRCUIT1\_ND

FROM idis.GCP\_NODE NODE2,

idis.GCP\_CIRCUIT CIRCUIT1

WHERE NODE2.NODEID=CIRCUIT1.STARTNODE

AND NODE2.NODETYPE='JUNIPER MX SERIES'

AND CIRCUIT1.CIRCUITTYPE LIKE 'ETHERNET CONNECTION%'

UNION

SELECT

NODE2.NODEID AS CIRCUIT1\_IPAGNODE,

CIRCUIT1.CIRCUITID,

CIRCUIT1.ENDLOCATION AS CIRCUIT1\_IPAGLOCATION,

CIRCUIT1.ENDNODE AS CIRCUIT1\_ND

FROM idis.GCP\_NODE NODE2,

idis.GCP\_CIRCUIT CIRCUIT1

WHERE NODE2.NODEID=CIRCUIT1.ENDNODE

AND UPPER(NODE2.NODETYPE)='JUNIPER MX SERIES'

AND UPPER(CIRCUIT1.CIRCUITTYPE) LIKE 'ETHERNET CONNECTION%'

) B3,

(

SELECT

NODE3.NODEID AS CIRCUIT2\_NTENODE,

NODE3.NODETYPE,

CIRCUIT2.CIRCUITID,

NODE3.DEVICESUBROLE,

CIRCUIT2.STARTLOCATION AS CIRCUIT2\_NTELOCATION,

CIRCUIT2.ENDNODE AS CKTND,

CIRCUIT2.NAME,

CIRCUIT2.AAFDAROLE

FROM

idis.GCP\_NODE NODE3,

idis.GCP\_CIRCUIT CIRCUIT2

WHERE

NODE3.NODEID=CIRCUIT2.STARTNODE

AND (

upper(node3.NodeType) in ('ADTRAN 800 SERIES','CISCO 900 SERIES') or

(upper(node3.NodeType) in ('CIENA 3900 SERIES','CIENA 8700 SERIES') and upper(node3.deviceRole) = 'NTE')

Or

(upper(node3.NodeType) = 'CIENA EMUX' and upper(node3.devicesubrole) = 'EMT')

)

AND CIRCUIT2.CIRCUITTYPE IN ('TIRKS BEARER','UNROUTED ETHERNET BEARER','ETHERNET BEARER')

UNION

SELECT

NODE3.NODEID AS CIRCUIT2\_NTENODE,

NODE3.NODETYPE,

CIRCUIT2.CIRCUITID,

NODE3.DEVICESUBROLE,

CIRCUIT2.ENDLOCATION AS CIRCUIT2\_NTELOCATION ,

CIRCUIT2.STARTNODE AS CKTND,

CIRCUIT2.NAME,

CIRCUIT2.AAFDAROLE

FROM

idis.GCP\_NODE NODE3,

idis.GCP\_CIRCUIT CIRCUIT2

WHERE

NODE3.NODEID=CIRCUIT2.ENDNODE

AND (

upper(node3.NodeType) in ('ADTRAN 800 SERIES','CISCO 900 SERIES') or

(upper(node3.NodeType) in ('CIENA 3900 SERIES','CIENA 8700 SERIES') and upper(node3.deviceRole) = 'NTE')

Or

(upper(node3.NodeType) = 'CIENA EMUX' and upper(node3.devicesubrole) = 'EMT')

)

AND CIRCUIT2.CIRCUITTYPE IN ('TIRKS BEARER','UNROUTED ETHERNET BEARER','ETHERNET BEARER')

) A5 ,

idis.GCP\_NUMBEROBJECTS NUMOBJS ,

idis.GCP\_NUMBER NUMBER1 ,

idis.GCP\_LOCATION LOCA8,

idis.GCP\_LOCATION LOCB8,

idis.GCP\_CLOUDOBJECTS CLOUDOBJECTS,

idis.GCP\_CLOUD CLOUD--,

--GCP\_SERVICE\_EXT SVC\_EXT

WHERE

SERVICEOBJECTS.SERVICEID = LEG.SERVICEID

AND SERVICEOBJECTS.RELATEDOBJECTTYPE = 'CIRCUIT'

AND A3.CIRCUITID = SERVICEOBJECTS.RELATEDOBJECTID

AND B3.CIRCUITID = SERVICEOBJECTS.RELATEDOBJECTID

AND A5.CKTND = A3.CIRCUIT1\_MDFNODE

AND NUMOBJS.RELATEDOBJECTTYPE = 'NODE'

AND NUMOBJS.RELATEDOBJECTID = A5.CIRCUIT2\_NTENODE

AND NUMBER1.NUMBERID = NUMOBJS.NUMBERID

AND NUMBER1.NUMBERTYPE in ('ADMIN ETHERNET IP ADDRESS','IPV4 TWAMP IP ADDRESS')

AND NUMBER1.NAME LIKE '%.%'

AND LOCA8.LOCATIONID = B3.CIRCUIT1\_IPAGLOCATION

AND LOCB8.LOCATIONID = A5.CIRCUIT2\_NTELOCATION

AND CLOUDOBJECTS.RELATEDOBJECTID = B3.CIRCUIT1\_IPAGNODE

AND CLOUDOBJECTS.RELATEDOBJECTTYPE = 'NODE'

AND CLOUDOBJECTS.CLOUDID = CLOUD.CLOUDID

AND CLOUD.TYPE = 'LATA'

AND EVC.SUBSCRIBERID=SUBSCRIBER.SUBSCRIBERID

--AND SVC\_EXT.SERVICEID = EVC.SERVICEID

AND UNI.SERVICETYPE='UNI SERVICE'

AND UNI.PRODUCTTYPE IN ('OEM-AC','SDN-ETHERNET')

AND EVC.SERVICETYPE='EVC SERVICE'

AND LEG.SERVICETYPE='LEG EVC SERVICE'

AND EVC.SERVICEID=EVC\_LEG.SERVICEID

AND EVC\_LEG.RELATEDOBJECTID= LEG.SERVICEID

AND EVC\_LEG.RELATEDOBJECTTYPE='SERVICE'

AND LEG.SERVICEID=LEG\_UNI.SERVICEID

AND LEG\_UNI.RELATEDOBJECTID= UNI.SERVICEID

AND LEG\_UNI.RELATEDOBJECTTYPE='SERVICE'

AND A5.NAME = UNI.NAME

AND LEG."In-Effect\_Indicator" = 'Y'

AND UNI.SUBSCRIBERID = UNI\_SUBSCRIBER.SUBSCRIBERID (+)

AND EVC.SERVICEID = EXT.SERVICEID (+);

**Initial Load:**

A one-time load needs to be sent across to the client (ATTeBiz) for each of the Organization (CUSTOMER) identified above.

In order to ensure there are no fallouts due to the file size which is restricted to <1MB, we can send the data in sequence. First at Organization (Customer) level, then at Asset (CUSTOMER\_NETWORK) level and then at asset (Account) level.

Meaning, the scenarios 1(Complete New BC Org Registered With ASE/ASEoD), 2(Addition of an EVC to the BC Org), 3(Addition of a New UNI to the existing EVC) from the attached file can be used as reference.

*Please note that, as we have huge data to be sent in initial load, based on the discussion with the client, we can start sending the data few weeks earlier than the deployment time (1902). Dev and ST teams to co-ordinate and make this work so that the data would be ready in client’s environment at the time of deployment.*

****

**Delta Load:**

The delta load is dependent on the changes being identified at each level of Organization(CUSTOMER), organization(Account), asset(CUSTOMER\_NETWORK), asset(ACCESS\_CIRCUIT) or customerAssetAlias.

The scenarios and corresponding JSON structure which would be applicable are mentioned in the attached file.

It also contains the processing logic which will be applicable for each of the scenarios. But dev team need to come up with the actual logic. This could be used as reference only.



All the changes can be identified by looking up in the change\_tracking.change\_timestamp.

**Insert/Update Scenario:**

A change can be identified in any of the following scenarios and the same would be reflected in the new tables – cust\_org\_details, customer\_network\_details, access\_circuit\_details.

* The change at Organization(Customer) can be identified by looking at the change\_tracking.change\_timestamp as it would be updated during the createOrg or updateOrg happens.
* The change in Organization(Account) which is associated with either Organization(Customer) or Asset.

Organization(Customer) <- Organization(Account) <- (CONTRACTED\_BY) <- Asset

* The change in the asset to asset associations

Asset(Access\_Circuit) -> Asset(Network\_Connection) -> Asset(Customer\_Network)

* The change in the contract associated with the Organization(Account)

Organization(Account) -> (CREATED\_FOR) -> Facilitation\_Contract

* The change in the alias\_association table which has the association between an Asset and the Alias\_Value.
* The change can also be in the attributes that are loaded from the source which can be tracked with the update in the current timestamp.

The event combinations which can be seen in each of the scenarios is given in the attached spreadsheet. This gives clear picture as what actionIdentifier would come in what scenario.



**Delete Scenario:**

In case of delete, we need to ensure that the following scenarios are taken care:

* A delete at an asset level can be identified by the change or removal of an association between the following

Asset(Access\_Circuit) -> Asset(Network\_Connection) -> Asset(Customer\_Network)

So if any association is removed that day, then the table should be updated with the corresponding actionIdentifier.

1. If an EVC is deleted, then return the following structure and post it to the topic.

{

"customer": {

"idOrganization": "524210151",

"actionIdentifier": "N"

},

"evc": [

{

"info": {

"assetUId": "550861736",

"actionIdentifier": "D"

}

}

]

}

You can first identify the EVCs that were deleted by using the following logic:

select asset.id

from metadata.source\_key\_deletion skd, gdb.asset, gdb.asset\_type, gdb.change\_tracking ct, gdb.change\_system cs

where skd.deletion\_timestamp > sysdate - 1

and skd.id\_source\_key = asset.id

and asset.id\_change\_tracking = ct.id and ct.id\_change\_system = cs.id and cs.name='IDIS\_TO\_GDB'

and asset.id\_asset\_type = asset\_type.id and asset\_type.type = 'CUSTOMER\_NETWORK';

Then you can identify the Organization(Customer) under which this is present by using the following logic:

select ac.assetId, ac.acctId, cust.id

from gdb.organization cust, gdb.association acct2cust, gdb.organization\_type ot,

(select assetId, acctId from (

select distinct asset.id assetId, acct.id acctId

from gdb\_hist.asset, gdb.organization acct, gdb\_hist.association asset2acct, gdb.change\_tracking ct

where asset2acct.id\_object\_what = asset.id and asset2acct.id\_object\_to = acct.id

and asset2acct.id\_change\_tracking = ct.id and ct.change\_timestamp > (sysdate - 1)

and asset.id in (

select asset.id

from metadata.source\_key\_deletion skd, gdb.asset, gdb.asset\_type, gdb.change\_tracking ct, gdb.change\_system cs

where skd.deletion\_timestamp > sysdate - 1

and skd.id\_source\_key = asset.id

and asset.id\_change\_tracking = ct.id and ct.id\_change\_system = cs.id and cs.name='IDIS\_TO\_GDB'

and asset.id\_asset\_type = asset\_type.id and asset\_type.type = 'CUSTOMER\_NETWORK'

)

UNION

select distinct asset.id assteId, acct.id acctId

from gdb.asset, gdb.organization acct, gdb.association asset2acct

where asset2acct.id\_object\_what = asset.id and asset2acct.id\_object\_to = acct.id

and asset.id in (

select asset.id

from metadata.source\_key\_deletion skd, gdb.asset, gdb.asset\_type, gdb.change\_tracking ct, gdb.change\_system cs

where skd.deletion\_timestamp > sysdate - 1

and skd.id\_source\_key = asset.id

and asset.id\_change\_tracking = ct.id and ct.id\_change\_system = cs.id and cs.name='IDIS\_TO\_GDB'

and asset.id\_asset\_type = asset\_type.id and asset\_type.type = 'CUSTOMER\_NETWORK'

)

)) ac

where acct2cust.id\_object\_what = ac.acctId and acct2cust.id\_object\_to = cust.id

and cust.id\_organization\_type = ot.id and ot.type = 'CUSTOMER\_ORGANIZATION\_REPRESENTATION';

1. If a UNI is deleted, then return the following structure

{

"customer": {

"idOrganization": "524210151",

"actionIdentifier": "N"

},

"evc": [

{

"info": {

"assetUId": "550861736",

"actionIdentifier": "N",

},

"accessCircuit": [

{

"info": {

"assetUId": "553970204",

"actionIdentifier": "D",

}

},

{

"info": {

"assetUId": "553970208",

"actionIdentifier": "D",

}

}

]

}

]

}

You should first retrieve the UNIs that are deleted by using the following logic:

select asset.id

from metadata.source\_key\_deletion skd, gdb.asset, gdb.asset\_type, gdb.change\_tracking ct, gdb.change\_system cs

where skd.deletion\_timestamp > sysdate - 1

and skd.id\_source\_key = asset.id

and asset.id\_change\_tracking = ct.id and ct.id\_change\_system = cs.id and cs.name='IDIS\_TO\_GDB'

and asset.id\_asset\_type = asset\_type.id and asset\_type.type = 'ACCESS\_CIRCUIT'

Then you can identify for which Organization(Customer) this is related to by looking into GDB\_HIST table if not present in the GDB schema by using the following associations:

ASSET (uni from above) 🡪 (USED\_BY) 🡪 ASSET (NETWORK\_CONNECTION) 🡪 (PART\_OF) 🡪 ASSET (CUSTOMER\_NETWORK)

ASSET (CUSTOMER\_NETWORK) 🡪 (CONTRACTED\_BY) 🡪 ORGANIZATION (Account) 🡪 (ROLLS\_UP\_TO) 🡪 ORGANIZATION (CUSTOMER\_ORGANIZATION\_REPRESENTATION)

**DMaaP Event Handler:**

1. SLAReportingEvent JSON is a Organization(CUSTOMER) specific and derived by the logic which ETL implement.
2. This event will check if the Organization(CUSTOMER) is present or not in the JSON structure. Here customer.idOrganization(1..1), customer.actionIdentifier(1..1) act as the rootKey elements which would be present in all scenarios of the JSON response.
3. If the customer.actionIdentifier is ‘C’, then treat this event as ‘create’. If the customer.actionIdentifier is ‘U’ or ‘N’, then treat this event as ‘update’. If the customer.actionIdentifier is ‘D’, then treat this event as ‘delete’.
4. If the root elements are not present, then do not generate any message but throw a standard error message.

## **Alternative Designs**

N/A

## **Assumptions/Risks**

## **Pre-Production Disaster Recovery Planning**

GCP currently meets AT&T Disaster Recovery Standards. GCP has 2 environments in production field, one is for Primary and one is for Disaster Recovery. GCP supports a transparent failover with little to no downtime.

## **Other Plans and References**

NA

|  |
| --- |
|  |

## **Acceptance & Approvals**

Overview

Use this section to capture approvals if electronic approvals via the PRISM Project Workflow Module will not be used.

The Approvers of this work product agree that this document is acceptable and complete to the best of their knowledge and will be used by the project team as an official deliverable for the project. It is further agreed that this document can now be baselined and any changes to these sections from this point forward must follow the Managing Change in the Technology Development Unified Process.

Embed evidence of approval in the review table below, or use the PRISM Approval Functionality in the Project Workflow Module Workflow Template View.

|  |  |  |  |
| --- | --- | --- | --- |
| **ATTUID and Name** | **Role** | **Group/Application** | **Version Approved, Approval Date and Approval Evidence** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |