Data Upload Use Case Scenarios

Scenario: Upload data in patient data object XML

The data load request message can be divided into three sections.

1. The <input_list> specifies the location of the PDO document.
2. The <load_list> specifies the PDO sections to load.
3. The <output_list> specifies the output expected from the upload process.

In addition, certain dependencies exist for loading some section of the PDO. The following table shows the dependencies for loading the individual PDO section.

Load PDO section	Dependent PDO section
event_set	eid_set
patient_set	pid_set
observation_set	eid_set, pid_set, event_set, patient_set
concept_set, modifier_set, observer_set, eid_set, pid_set	NONE

The example shown below contains the sections of the PDO XML that are required to load a single observation fact.

- **Note**
  The dependency section in the PDO is optional; if they exist in the data mart

For a more detailed example on loading patient and encounter mapping, please refer to the *CRC Design Document*.

```
<patient_data>
<observation_set>
<observation update_date="2006-05-04T18:13:51.0Z"
sourcesystem_cd="TEST">
<event_id source="MGHE">MENT001</event_id>
<patient_id source="MGHP">MPAT001</patient_id>
<concept_cd name="C">i2b2CD</concept_cd>
<observer_cd source="TEST">@</observer_cd>
<start_date>2006-05-04T18:13:51.0Z</start_date>
<instance_num>1</instance_num>
<modifier_cd>modifier_cd</modifier_cd>
<valuetype_cd>valuetype_cd0</valuetype_cd>
<nval_num units="units0">3.141592653589</nval_num>
<valueflag_cd name="name8">valueflag_cd0</valueflag_cd>
<quantity_num>3.141592653589</quantity_num>
<confidence_num>3.141592653589</confidence_num>
</observation>
</observation_set>
<pid_set>
<pid>
<patient_id status="A" update_date="2006-05-04T18:13:51.0Z"
sourcesystem_cd="TEST" source="MGHP">MPAT001</patient_id>
</pid>
</pid_set>
<eid_set>
<eid>
<event_id source="MGHE" patient_id="MPAT001"
patient_id_source="MGHP" status="A"
update_date="2006-05-04T18:13:51.0Z"
sourcesystem_cd="TEST">MENT001</event_id>
</eid>
</eid_set>
<event_set>
<event update_date="2006-05-04T18:13:51.0Z"
sourcesystem_cd="TEST">
<event_id source="MGHE">MENT001</event_id>
<patient_id source="MGHP">MPAT001</patient_id>
<start_date>2006-05-04T18:13:51.0Z</start_date>
<end_date>2006-05-04T18:13:51.0Z</end_date>
```

```
</event>
</event_set>
</patient_data>
```

Request Message:

```
<message_body>
<publish_data_request>
<input_list>
<data_file>
<location_uri protocol_name="FR|LOCAL">location_uri0</location_uri>
<data_format_type>PDO</data_format_type>
<source_system_cd>source_system_cd0</source_system_cd>
<load_label>load_label0</load_label>
</data_file>
</input_list>
<load_list commit_flag="true" clear_temp_load_tables="false">
<load_observation_set ignore_bad_data="true"/>
<load_event_set ignore_bad_data="true"/>
<load_patient_set ignore_bad_data="true"/>
<load_eventid_set ignore_bad_data="true"/>
<load_pid_set ignore_bad_data="true"/>
<load_eid_set ignore_bad_data="true"/>
<load_concept_set encrypt_blob="false" ignore_bad_data="true" delete_existing_data="true|false"/>
<load_modifier_set encrypt_blob="false" ignore_bad_data="true" delete_existing_data="true|false"/>
<load_observer_set ignore_bad_data="true" delete_existing_data="true|false"/>
</load_list>
<output_list detail="true">
<observation_set onlykeys="true" blob="false" techdata="false"/>
<patient_set onlykeys="true" blob="false" techdata="false"/>
<event_set onlykeys="true" blob="false" techdata="false"/>
<observer_set onlykeys="true" blob="false" techdata="false"/>
<concept_set onlykeys="true" blob="false" techdata="false"/>
<modifier_set onlykeys="true" blob="false" techdata="false"/>
<pid_set onlykeys="true" blob="false" techdata="false"/>
<eid_set onlykeys="true" blob="false" techdata="false"/>
<eventid_set onlykeys="true" blob="false" techdata="false"/>
</output_list>
</publish_data_request>
</message_body>
```

Element Name	Description
<input_list>	**This section of xml will carry input information such as the location of the PDO file, its data format, etc.**
location_uri	The location of the input PDO file.
protocol_name	Protocol name that specifies the service for how to access the PDO file in the location_uri. FR – File Repository LOCAL – File resides in the server's local folder
data_format_type	PDO (Patient Data Object XML)
source_system_cd	Upload's the code for the source system
load_label	The user's load label
<load_list>	**This section of the request holds the load options.**
commit_flag	The current implementation supports only a value of "true". This is a placeholder for future releases. If the flag is set to false, the system will just run all the load process for the given PDO data and will not perform commit. This is a useful option to check, whether the given PDO data, have the valid information.
clear_temp_load_tables	This flag specifies whether to clean up the staging area. Set this flag to false when you need to debug.
ignore_bad_data	Currently this is not implemented. It will be part of a future release.
load_observation *set,* *append_flag*	Load or update the fact entry coming from the input PDO data. If the append_flag = 'true' is specified then the PDO's fact information is just added to the stored fact and will not do any updates.
load_event_set	The information from the PDO's <event_set> is used to load the data to the VISIT_DIMENSION table.

load_patient_set	The information from the PDO's <patient_set> is used to load the data to the PATIENT_DIMENSION table.
load_observer_set *set* *delete_existing_data*	The information from the PDO's <observer_set> is used to load the data to the PROVIDER_DIMENSION table.
load_pid_set	The information from the PDO's <pid_set> is used to load the data to the PATIENT_MAPPING table.
load_eid_set	The information from the PDO's <eventid_set> is used to load the data to the ENCOUNTER_MAPPING table.
load_concept_set *set* *delete_existing_data*	The information from the PDO's <concept_set> is used to load the data to the CONCEPT_DIMENSION table. Set the delete_existing_flag to remove the old data before loading the new data and the old data will be available in the back up table.
load_modifier_set *set* *delete_existing_data*	The information from the PDO's <modifier_set> is used to load the data to the MODIFIER_DIMENSION table. Set the delete_existing_flag to remove the old data before loading the new data and the old data will be available in the back up table.
<output_list>	This section of the request is used for getting the upload process status information.
detail	This attribute is a placeholder for a future release. The attribute specifies whether the response message should have detailed information. If the value is false, then the service will return only the count for the number of inserted and updated records. Otherwise a separate file with the inserted and updated records will be created.
observation_set	This element specifies the process to fetch the details of updated records on the OBSERVATION_FACT table.
onlykeys	Return the primary key fields in the response
blob	Return the blob field in the response
techdata	Return the techdata fields in the response
patient_set	The element specifies the process to fetch the details of updated records on the PATIENT_DIMENSION table.
event_set	The element specifies the process to fetch the details of updated records on the VISIT_DIMENSION table.
observer_set	The element specifies the process to fetch the details of updated records on the PROVIDER_DIMENSION table.
concept_set	The element specifies the process to fetch the details of updated records on the CONCEPT_DIMENSION table.
modifier_set	The element specifies the process to fetch the details of updated records on the MODIFIER_DIMENSION table.
pid_set	The element specifies the process to fetch the details of updated records on the PATIENT_MAPPING table.
eid_set	The element specifies the process to fetch the details of updated records on the ENCOUNTER_MAPPING table.

Response Message:

```
<message_body>
<load_data_response>
<status>
<condition type="ERROR|DONE" coding_system="">
error message
</condition>
</status>
<upload_id>upload_id0</upload_id>
<user_id>user_id0</user_id>
<data_file_location_uri protocol_name="FR|LOCAL">
location_uri0
</data_file_location_uri>
<load_status>load_status0</load_status>
<transformer_name>transformer_name0</transformer_name>
<start_date>2006-05-04T18:13:51.0Z</start_date>
<end_date>2006-05-04T18:13:51.0Z</end_date>
<message>message0</message>
<observation_set inserted_record="0" ignored_record="0" total_record="0">
<ignored_patient_data_file_uri protocol_name="FR|LOCAL">
uri0
</ignored_patient_data_file_uri>
<message>message1</message>
</observation_set>
<patient_set inserted_record="0" ignored_record="0" total_record="0">
<ignored_patient_data_file_uri protocol_name="FR|LOCAL">
</ignored_patient_data_file_uri>
<message>message2</message>
</patient_set>
<event_set inserted_record="0" ignored_record="0" total_record="0">
</event_set>
<observer_set inserted_record="0" ignored_record="0" total_record="0">
</observer_set>
<concept_set inserted_record="0" ignored_record="0" total_record="0">
</concept_set>
<modifier_set inserted_record="0" ignored_record="0" total_record="0">
</modifier_set>
<pid_set inserted_record="0" ignored_record="0" total_record="0">
</pid_set>
<eventid_set inserted_record="0" ignored_record="0" total_record="0">
</eventid_set>
</load_data_response>
</message_body>
```

Scenario: Get upload status info by upload_id and user_id

This message will retrieve one of the following:

1. A list of upload status information based on the user_id
2. A single upload status by upload_id

Request Message:

```
<message_body>
<get_upload_info_request>
<user_id>user_id</user_id>
<upload_id>100</upload_id>
</get_upload_info_request>
</message_body>
```

Response Message:

```
<message_body>
<load_data_list_response>
<load_data_response>
Please refer section 4.5.1.1 for <load_data_response/>
</load_data_response>
. . .
</load_data_list_response>
</message_body>
```

Scenario: Find the unmapped term in the datamart

This service returns a count for the number of concepts, modifiers and providers that considered "unmapped". A term is "unmapped" when it is found in the fact table and is missing from the corresponding dimension table.
The unmapped term can be requested for a particular upload or for the entire data mart. If detail="true" then the service will return the list of unmapped terms.

Request Message:

```
<message_body>
<get_missing_term_request upload_id="NN" detail="true | false">
<concept_set start="1" end="1000" />
<modifier_set start="1" end="1000" />
<observer_set start="1" end="1000" />
</get_missing_term_request>
</message_body>
```

Response Message:

```
<message_body>
<missing_term_report_response>
<missing_term_report upload_id="22">
<concept_set mapped="41683182" unmapped=""/>
<modifier_set mapped="41683182" unmapped="" />
<observation_set mapped="41683182" unmapped=""/>
</missing_term_report>
<missing_codes>
<concept_set>
<concept missing_total="9">
<concept_cd>brine:ses5</concept_cd>
</concept>
<concept missing_total="5">
<concept_cd>MTP:KGC</concept_cd>
</concept>
</concept_set>
<observer_set>
<observer/>
</observer_set>
<modifier_set>
<modifier/>
</modifier_set>
</missing_codes>
</missing_term_report_response >
</message_body>
```