

U N I V E R S I T Y O F H A W A I I ' I A T M Ā N O A

Institute for Astronomy

Pan-STARRS Project Management System

**Pan-STARRS PS1 Image Processing Pipeline-Published Science
Products Subsystem
Interface Control Document Version 2.0
(PS1 IPP-PSPS ICD V2.0)**

Conrad Holmberg
Software Engineer
University of Hawaii
24 November 2009

PSDC-300-xxx-DR (properties, custom to set)

© Institute for Astronomy, University of Hawaii
2680 Woodlawn Drive, Honolulu, Hawaii 96822
An Equal Opportunity/Affirmative Action Institution

Revision History

Version/Revision	Date	Comments
DR	4/6/2009	Creation of report
2		Release of report

Table of Contents

1 Scope of Document.....	3
2 Referenced Documents.....	3
3 Introduction.....	4
4 Process Overview.....	4
5 IPP Data Publication Process.....	5
5.1 IPP Job Manifest Description.....	7
5.2 IPP Batch Manifest.....	8
5.2.1 Initialization Batch.....	9
5.2.2 Detection Batch.....	9
5.2.3 Stack Batch.....	9
5.2.4 Difference Batch.....	10
5.2.5 Object Batch.....	10
5.3 IPP FITS Files.....	11
5.4 IPP File Server.....	12
6 Error Handling of IPP Publications.....	13
6.1 Job Refused Error.....	13
6.2 Job Manifest Error.....	14
6.3 Job Content Error.....	14
6.4 Batch Manifest Error.....	14
6.5 Batch Content Error.....	15
6.6 Batch Load Error.....	15
7 Error Handling Protocol of IPP Data Publications.....	15
7.1 Error Messages.....	15
7.2 Error Handling Resolutions.....	16
7.3 The IPP Recovery.....	16

7.4 Repair.....	17
7.5 Cancellation.....	17
8 Definitions.....	17
9 Naming Conventions.....	17

List of Figures

Figure 1. A general Data Flow Diagram (DFD) for the entire ICD for PS1.....	5
Figure 2. A general DFD for the IPP Publication.....	6
Figure 3. Job Manifest File Example.....	7
Figure 4. Batch Manifest File Example.....	8
Figure 5. Metadata definition example of PSPS database schema.....	12
Figure 6. Data Set Publication Example.....	13
Figure 7. Example of Data Store Server for Error messages.....	15

List of Tables

Table 1. PSDC Documents.....	3
Table 2. External Documents.....	4
Table 3. Table of Error Handling Resolutions.....	16

1 Scope of Document

The purpose of this document is to give a detailed description for the Image Processing Pipeline (**IPP**) on how to create science publications for the Published Science Product Subsystem (**PSPS**). Note that this is a total revision of the original Interface Control Document (**ICD**) (hence version 2.0) which is in some ways very consistent but with more details on TBD factors and other unknowns. However, there are still unresolved issues in this version as well in the hopes that it will be changed continually to find solutions.

2 Referenced Documents

Table 1. PSDC Documents

Pan-STARRS ID	Title	Authors
PS1 IPP-PSPS ICD	Pan-STARRS PS1 Image Processing Pipeline-Published Science Products Subsystem Interface Control Document	Heasley, Magnier
NA	DX-State Diagram	Holmberg
NA	DX-LoadDB Workflow	Catharine Van Ingen

Pan-STARRS ID	Title	Authors
NA	Mapping FITS ODM	Heasley
NA	DX-LoadDB Workflow, On Tables and Batches	Van Ingen

Table 2. External Documents

Source Reference	Title	Authors
http://heasarc.gsfc.nasa.gov/FITSio/	CFITSIO	HEASARC Greenbelt, MD
http://en.wikipedia.org/wiki/Md5sum	Md5Sum	Wikipedia article

3 Introduction

The ICD version 2.0 has the intentions of covering requirements that have already been noted and resolved but in more detail or give details to new requirements that were not anticipated beforehand. Some of this document will be a repeat to the original ICD, but keep note of changed definitions and other requirements such as error definitions and data recovery scenarios.

The main goal for Pan-STARRS from the scope of the ICD is for the IPP to publish science data in such a way that a component of the PSPS called the Data Transformation Layer (DXLayer) can convert this data into something that can be loaded by the ODM (Object Data Manager) component and finally into a PSPS database that will be published to outside users. This ICD documents the details of this which includes naming conventions, error handling, and data repair.

4 Process Overview

The entire process of taking a science publication from the IPP and loading it into a PSPS database involves three main components:

- IPP Data Publication – The IPP has taken science data, convert it into a readable format, compressed it, and loaded it onto a file server called the Data Store Server for downloading.
- DXLayer – The DXLayer from the PSPS has found a new publication with compressed files, continues to download, read, convert, and send them for Loading.
- ODM – A message is received from the DXLayer for new data to be loaded and the workflow process begins.

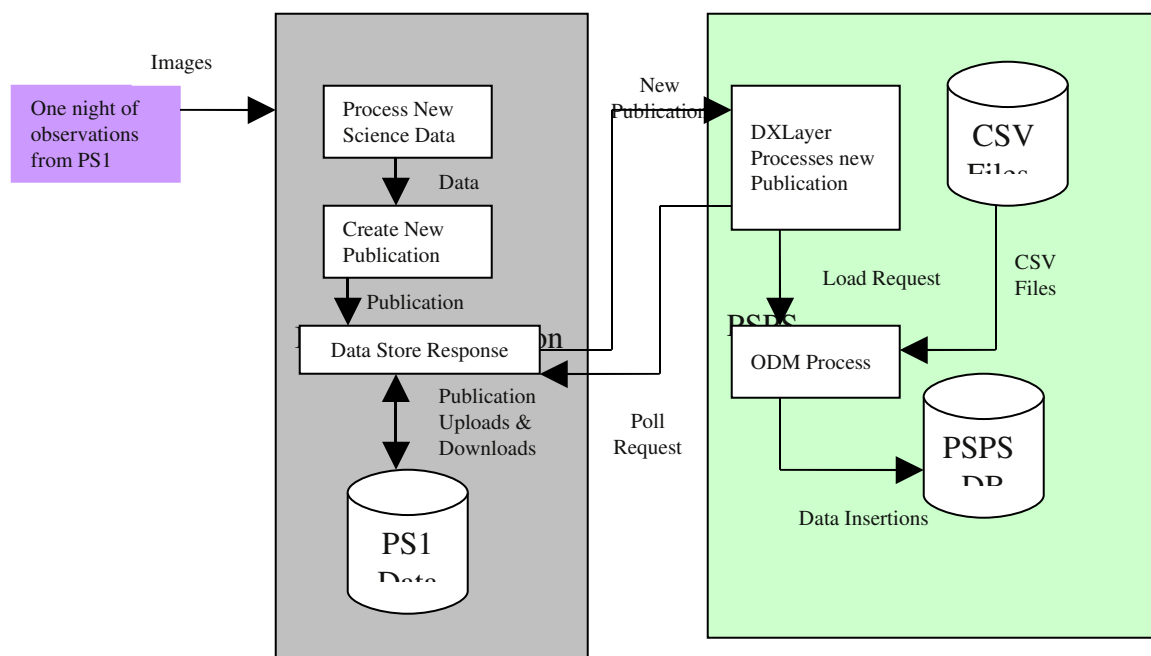


Figure 1. A general Data Flow Diagram (DFD) for the entire ICD for PS1.

The Data Flow Diagram in Figure 1 is a very general overview of these components with the ICD. The functionality between the DXLayer and the ODM and how published science is placed on a PSPS database is beyond the scope of this document. The main focus is to describe the structure of the publication, details of messaging, descriptions of the errors, and how to implement repairs.

5 IPP Data Publication Process

The IPP Data Publication process can be viewed as serving three major purposes:

- Publication of the Science Data from Telescope images.
- Providing availability of these publications via file server to the PSPS.
- Account for error handling and providing repaired publications (explained in the next section).

An IPP science publication of data that is loaded onto a file server constitutes as a “Job”. A Job is defined as the equivalent of “**one night’s worth of observations**” which is composed of several batches. Each Job will have a manifest file that describes each Batch. The naming convention of a Job is identified with a J and a six digit number with zero padding (i.e., J000001).

A “Batch” consists of a manifest and FITS files containing binary tables that describe the actual data that will be loaded into the PSPS database. The Batch manifest file contains metadata that describes

each FITS file within the Batch of one frame (**TBD Define the batch as one frame**). The naming convention of a Batch is identified with a B and a three digit number with zero padding (i.e., B001) that increments after every batch.

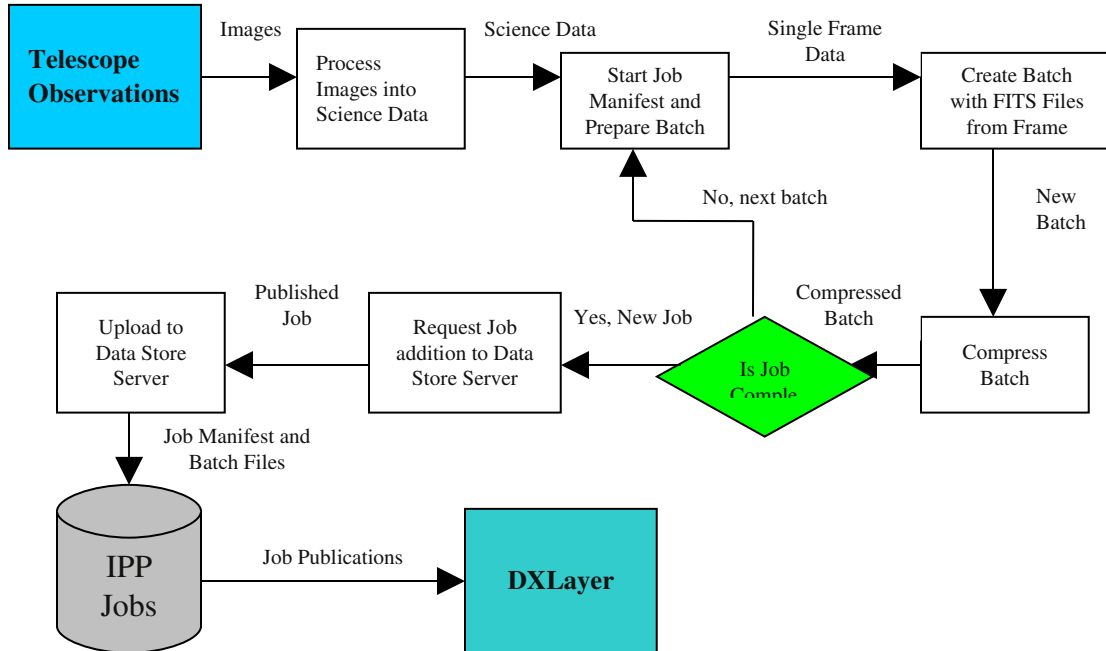


Figure 2. A general DFD for the IPP Publication.

The Data Flow Diagram above in Figure 2 shows a very brief layout of the publication (**Note: Yes this looks very juvenile, but we don't have to detail this out**). There is a lot of work and sub-processes within this of course, but it's only meant as a general overview.

The IPP publication simply starts with one night's worth of observations. For that night, one Job Identification number is given. Once all the images have been transformed into science data, a Job manifest is started and a single frame is processed into a group of FITS files depending on the given data, survey, type, etc (**TBD**). Once all the FITS files are complete, it is compressed as a Batch with a manifest describing it. If it is the last Batch within the Job; then the Job is requested to be uploaded to the Data Store Server along with its manifest. If not, then the next batch is done in the loop until completion. Once a Job publication has finished and completely uploaded to the Data Store Server; then it is available to the DXLayer for preparing for the ODM.

5.1 IPP Job Manifest Description

The Job manifest consists of a metadata that will contain several characteristics that PSPS needs in order to load data, this includes:

- A Job identification number denoted as the "name".
- The survey identification that is noted by the "type".
- A timestamp indicating when the publication was created
- The list of batches each with a name, size, and checksum.

An example of a Job Manifest file in XML is show below:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE manifest SYSTEM "psps-manifest.dtd">
<manifest name="J000001" type="3PI" timestamp="2009-04-07 21:39:31">
  <file name="B001.tar.gz" bytes="1002384" md5="xxx"/>
  <file name="B002.tar.gz" bytes="100789" md5="yyy"/>
</manifest>
```

Figure 3. Job Manifest File Example

Each batch file within the Job manifest is a compressed group of FITS files. The format must be a tarball gzipped compression and the directory that it unpacks to must be the same as the batch name. So B001.tar.gz must uncompress a directory called B001 that contains a manifest and all the FITS files. The naming convention for the Job Manifest file is simply JobManifest.xml (**TBD whether to add the Job ID to the manifest file name**) which is consistent for the DXLayer to be able to know what to download first. If the administrator is unsure of where the manifest belongs, the Job Identification number is contain within the file.

Please note that the Survey Identification is needed for loading purposes. Also this could be indentified as the initial load.

TODO: Define the survey and explain its purpose

The checksum for each Batch will be calculated by an m5sum algorithm and use 128-bit MD5 hashes and the file size displayed in bytes.

5.2 IPP Batch Manifest

A Batch Manifest consists of an xml file that and several FITS files that contain binary tables of the actual science data that the PSPS needs to be loaded. A Batch has the following characteristics:

- A batch identification number denoted as the “name”.
- The batch type that is noted by the “type” (see 5.2.1-5.2.5).
- A timestamp indicating when the batch was created
- The list of FITS files each with a name, size, and checksum.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE manifest SYSTEM "psps-manifest.dtd">
<manifest name="B001" type="P2" timestamp="2009-04-07 21:39:35">
  <file name="F01_p2.FITS" bytes="100238" md5="xxx"/>
  <file name="F02_p2.FITS" bytes="100789" md5="yyy"/>
</manifest>
```

Figure 4. Batch Manifest File Example

One important aspect of the batch is the type which can be:

- Initialization – The initial load from the IPP that populates the tables.
- Detections – Data containing objects encounters.
- Stack – New object encounters?
- Difference – Tracking encounters of objects?
- Objects – These are actual defined objects.

There are several things that can consist of a single batch. A batch can be generated per frame but this does not relate to all the types available such as an object Batch Type isn't necessary associated with a frame. **TBD – What distinguishes a batch? We still do not have a strong description here.**

Each FITS file within the batch must contain tables related to the batch type. For example, if the batch type is a detection type; then all FITS tables must adhere to the detections and not stacks or any other type.

The checksum for each FITS file will be calculated the same as the batches (see Md5sum reference).

The next section references each Batch Type as in the document “Notes on Tables and Batches”.

5.2.1 Initialization Batch

This is not strictly a “batch” from the IPP. There are a number of tables we need to get populated in the ColdDBs the very first time. How we do this is currently TBD. The tables included are:

- CameraConfig- Note that the primary key CameraID is assigned by the camera group.
- Filter.
- FitModel.
- PhotoCal.
- PhotozRecipe.
- SkyCell spline of SkyCellID to RegionID. 1:1 a Skycell is exactly one region.
- Survey.
- Region (TBD)
- PartitionMap This (not yet back in the schema) table contains
- PSConstants This (TBD, defined in schema?) contains the various constants we need such as the zone definition increment.

5.2.2 Detection Batch

One frame generates one Detection Batch which is loaded into one database. New objects may be encountered when loading a Detection Batch. In addition to the object table, each Detection Batch loads information into the following tables:

- P2FrameMeta.
- P2ImageMeta (N images per frame)
- P2Detection
- P2Orphan
 - There is no orphan file as the DXLayer will split these.

5.2.3 Stack Batch

New objects may be encountered when loading a stack batch. In addition to the object table, each stack batch loads information into the following tables:

- StackMeta
 - There is no consistency or correctness checking on the various IDs in this table other than that they exist. stackmetaid, skycellid,
- P2ImageToStack.
 - This is a many:1 spline. Each StackMetaId is associated with multiple ImageIds.
- StackDetection
- StackOrphan. This is optional.
- StackApFlx, StackB2DFit, StackModelFit. (These tables are optional).

5.2.4 Difference Batch

New objects are not encountered when loading a difference (or delta) batch. Each difference batch loads information into the following tables:

- StackMeta
- P2ImageToStack
- StackLowSigDelta
- StackHighSigDelta
 - This table is similar to the StackDetection or P2Detection.
- StackDeltaAltFit - keys with stackdetectID (key in StackHighSigDelta), ippDetectID, objID, modelID. (This is optional)

5.2.5 Object Batch

These batches fill in the statistics and object property columns of the object table. The only affected table is the Object table. These batches do NOT contain newly resolved objects. That happens only in a P2 or Stack batch.

Open questions/notes:

- The varFlag and ObjInfoFlag bit masks are currently TBD.
- The “Bar” columns (eg raBar, raBarVar) columns are the computed “mean” locations with proper motion removed. (This is not currently documented). These are computed

by the IPP because we don't have all of the necessary algorithms/constants for computing the proper motion. This is also in line with the "IPP does the computations, we do the persistence for queries" work split.

- Stackver is the stackDetection version (stackver) for the stack that sources the "best" science variables held later.

Ndetections is the number of detections used to do the positional information in the columns to the left.

5.3 IPP FITS Files

A data publication from the IPP will appear in a FITS file format with binary tables containing the actual data.

The naming convention of the FITS files will consist of:

FRAMEID[_NUM]_ FTYPE.EXT

The FTYPE is the enumeration for the frame types, the FRAMEID is a unique numeric identifier that is zero padded and the extension is FITS (i.e., F01_P2.FITS **TODO determine whether or not this naming convention suffices**). In the case that a FITS file becomes too large and must be split by incrementing the frame id number and make sure that it is in this order within the Batch manifest file!

The DXLayer needs a guide in order to determine how a FITS file relates to a MS-SQL database. The reason being is that FITS column name, data types and other characteristics are different from a conventional database. The solution to this problem is to compose a XML file containing a metadata structure of each table within the PSPS schema that defines both the MS-SQL and FITS relations. An example of how this can be defined is below:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE manifest SYSTEM "psps-schema.dtd">
<schema name="PSPS">
<!-- PS01 Table Schema -->
  <table name="P2FrameMeta" ext_name="P2FrameMeta" FITS_type="table">
    <column>
      <name>frameID</name>
      <type>BIGINT</type>
      <decimal_max>18</decimal_max>
      <decimal_min>0</decimal_min>
      <FITS_name>FRAMEID</FITS_name>
      <FITS_type>TLONG</FITS_type>
      <FITS_unit>NONE</FITS_unit>
      <input_type>insert</input_type>
      <comment>unique ID for each image, hashed from frameID</comment>
    </column></table>
  </manifest>
```

Figure 5. Metadata definition example of PSPS database schema

The original ICD described each binary table in relation to the schema. However, with a metadata description of the PSPS schema, then this can act as the guide for converting FITS into CSV files. Notice that there are separate tags for names and data types for FITS and MS-SQL. This is so the DXLayer knows the relationships and can check for inconsistencies (i.e., an extra or missing column in the FITS table that is not in the metadata).

The definitions of the tags can be explained as follows:

- **Table** – the attributes include the PSPS database table name and the FITS extname which is given within the HDU of the binary table. The FITS type is also given.
- **Column** – Columns have several tags both relating separately to FITS tables and the PSPS schema.
 - **Name** – This is PSPS schema name of the table column.
 - **Type** – This is the MS-SQL data type of the table column.
 - **Decimal Min and Max** – The minimum and maximum value applied to a number (NA to strings) for significant digits. For example, with a min value of 0.000 and a max of 9.000, the min is 3 and the max is 1. The ranges are not checked by the DXLayer, as this happens within the ODM
 - **FITS Name, Type, and Unit**- These tags will all be specified by the IPP in order to properly generate FITS files. (TBD Do we know to elaborate on this)
 - **Input Type** – This determines whether or not the IPP gives this within the FITS files or the DXLayer calculates it. (TBD Right now, the DXLayer does not calculate anything and the ODM does the RA and DEC so is this really necessary?)
 - **Comment** – Description of the column that is contained within the FITS HDU and the PSPS Schema.

Some general rules that apply to the structure of FITS files:

- Since all the FITS files are only concerned with data, then the Image HDU must not contain any data and is only there to consist of a valid FITS file.
- The HDU of a FITS binary table must have the EXTNAME reserved word for the name of the table that is in the metadata.
- Each column must be in the HDU with the data types and must match what is in the metadata.

TBD: Determine the maximum size for any batch or FITS file. Download speeds are going to be so fast that I honestly don't think that it will matter because if we are splitting things by the frame, but the size most probably will be under 5 Gigabytes in most situations.

5.4 IPP File Server

The IPP will provide a custom file server application called the “Data Store”. This is a simply set of CGI scripts that will provide the DXLayer published data via HTTP. The IPP

uploads a new Job to the Data Store as a file set that has the same name as the Job ID. This file set contains the Job manifest file and all of the associated batch files. An example of the URL path would be:

`http://ps.domain.org/ds/psps_data/J000001/index.txt`

The Job ID becomes part of the path with the new file set. The Data Store output of the new job would look like this:

#	fileID	bytes	md5sum	type
	JobManifest.xml	251	015e05dab5e6844b704129e6bd23ddd2	xml
	B001.tar.gz	1002384	befe205d241c6ca36659b9bbc0ee19cd	tgz

Figure 6. Data Set Publication Example

Using this format via the Data Store, the DXLayer can easily download IPP publications and do the required error checking to make sure that the file size and checksums match.

Note that a Job cannot be available to the DXLayer until ALL the batches within the manifest have been compressed to insure proper validation.

6 Error Handling of IPP Publications

The only communication between the IPP and the PSPS components is on the account of errors. There are no notifications for successful loads as it is assumed that the PSPS is functioning normally. When an error report is given, it will contain the following:

- Job ID
- Batch ID – Only added if applicable.
- Error Name – A list of these is given below.
- Description- This can include details like machine names, stack traces, and possible causes.

A comprehensive list of all the possible errors is described in each subsection below along with the scenarios on how it can happen and the actions required for a resolution. **TODO – We must examine each error to determine whether a recovery or repair is needed. Some are less obvious than others.**

6.1 Job Refused Error

Description: The PSPS cannot access the Data Store Server.

Examples: There are several reasons for this which can include: a network failure, the Data Store server is unavailable, or firewall security issue.

Action: The DXLayer stops loading data and sends a generated email to the systems administrator and the error is logged.

Resolution: If networking is no longer available between the IPP and PSPS components, manual intervention is most likely needed from a systems administrator. For instance if a restart of the server is needed; then the DXLayer can recovery from the last known successful job and download and convert all the batches again from that point.

6.2 Job Manifest Error

Description: An inconsistency exists between the Job Manifest and the Batch files.

Examples: The Data Store is missing a file that is listed within the Job Manifest or vise versa. This error can also occur if the checksum or file is consistent. This could also happen if an individual edits something and makes a mistake causing an inconsistency.

Action: The DXLayer stops loading data and sends a generated email to the systems administrator and the error is logged.

Resolution: It is the responsibility of the IPP to fix this within their components that produce the data publications. The does not require a repair version because this batch in particular has not been sent to the ODM for loading. The DXLayer is restarted from that Job once the repair has been made.

6.3 Job Content Error

Description: The Contents within the Job do not correspond with what the PSPS is expecting from an IPP science publication.

Examples: The database schema has changed within the PSPS and does not match an IPP FITS table or there is a mistake within calculations, data types, or ranges within the data. Another situation is when the Job ID already is a duplicate when it already has been completed previously.

Action: The DXLayer stops loading data and sends a generated email to the systems administrator.

Resolution: The IPP must fix the content within the publication and send a repair version of the bad Job. If it is decided that this Job has can nothing loading and has too many errors, then no repaired is made and the entire Job is cancelled.

6.4 Batch Manifest Error

Description: An inconsistency exists between the Batch Manifest and the FITS files.

Examples: The Data Store is missing a file that is listed within the Job Manifest or vise versa. This error can also occur if the checksum or file is consistent. This could also happen if an individual edits something and makes a mistake causing an inconsistency.

Action: The DXLayer stops loading data and sends a generated email to the systems administrator.

Resolution: TBD Determine if a repair version is needed since it is not sent to the ODM for loading.

6.5 Batch Content Error

Description: This happens when something is expected within the FITS file such as a column or a missing table, etc. The FITS has no errors (which is in that case a batch corruption error), but the content is either unexpected for missing.

Examples: A missing FITS file stated within the batch manifest is not within the uncompressed batch.

Action: The DXLayer stops loading data and sends a generated email to the systems administrator and the error is logged.

Resolution: A repair version of that Batch is published on the Data Store Server. (TBD: Are there going to be repair versions for the DXLayer)

6.6 Batch Load Error

Description: The Batch has been successfully sent to the ODM for loading, but something is wrong with the data and it cannot be loaded.

Examples: An expect column is out of range or missing.

Action: The DXLayer can continue to load and isn't even aware of this error. It is up to the administrator to stop loading if too many of these errors are occurring. **Note: The Job and Batch verification errors have are now deprecated because this is not known until the load process.**

Resolution: Since this error occurs on the ODM side, a repair Job must be start be created with each repaired batch file and ignore the batches that are fine.

7 Error Handling Protocol of IPP Data Publications

This section provides how errors are handled from messages to the different types of resolutions.

7.1 Error Messages

The DXLayer will send all messages to the IPP using the Data Store Server Error Messaging. This acts as the same mechanism as which the DXLayer downloads publications. The difference is that the DXLayer uploads an error

#	fileID	bytes	md5sum	number	type	status
J0001_B002.txt		251	xxx	123	BATCH_MANIFEST	closed
J0123_B001.txt		1002384	yyy	125	BATCH_CONTENT	open

Figure 7. Example of Data Store Server for Error messages.

The example above shows several columns similar to the IPP Data Store Server for publications. The extra columns are as follows:

- Number – The error number given the DXLayer
- Type – The type of error described in section 5.

- **Status** – The status is either open or closed determining if the bug has been fixed or not.

TBD – Do we need this? I am under the impression that the IPP must run totally independent of the PSPS. However, if errors are posted on the DXLayer database, they can be reached there, but the IPP needs to know them and the Data Store Server can do this. This issue must be resolved.

The IPP can poll this instance of a Data Store and handle errors as they are posted. The DXLayer can change the status as fixes are made.

7.2 Error Handling Resolutions

The IPP has three types of resolutions: Recovery, Repair or Cancellation. Each type of recovery is explained in the table below. Each of the errors will have a type of recovery, but this is somewhat arbitrary to the situation and administration decisions must be made.

Type	Description	Example
Recovery	The Job is repaired on the spot and the DXLayer is restarted to convert that Job over again	A brief network outage between the Data Store Server and the DXLayer.
Repair	The Job containing errors is published again as a “repair version” and the Batches that have found errors recreated and the DXLayer converts these again from the ODM	Data that was queued for loading failed because there are duplicates IDs.
Cancellation	The Job in question and all the batches within are deemed NFG and the entire Job is cancelled and no data is loaded.	Nothing has been loaded because the entire Job

Table 3. **Table of Error Handling Resolutions**

7.3 The IPP Recovery

A recovery is in the case of an external error where the Job can be processed over again by the DXLayer and continues on to the next Job. There is nothing wrong with the data, but if something like a network outage occurs and the DXLayer cannot download a Job, an error is reported and it shuts down. Once the DXLayer is restarted and continues from that Job.

7.4 Repair

In the event that Job has Batches containing errors. It is possible to do a repair on this Job. Constructing a repair Job is very easy. You simply append an R for repair to the Job ID and a version number and publish it again on the Data Store Server. When the DXLayer discovers that this is a repair, it will log this and convert the repair batches and send them to the ODM as a repair. Only the repair batches will be contained within the repair version.

For example, if the Job J03 contains batches B1, B2, and B3 and if B2 is the only batch containing errors, then the repair publication would be J03R with a single batch named B2R.

7.5 Cancellation

In the event that a Job has too many errors to consider it repairable, the entire Job can be cancelled. The notification of the cancellation is given via a Data Store Server Error Messaging. Once the DXLayer discovers this in a poll, it logs the event and deletes all its contents.

8 Definitions

Job – The equivalent of one night's worth of observations composed of several batches. Each Job will have a manifest file that describes each batch. A job is identified with a J and a six digit number with zero padding (e.g. J000001).

Job Manifest – A Job Manifest contains a manifest file and all the batches within a Job publication. It is in XML format and contains an entry that describes each Batch with a name, size, and checksum.

Batch - A batch is a compressed file that contains FITS files of a certain type. Batch Types include Objects, Stack Detections, and P2 Detections.

Batch Manifest – A Batch manifest contains a manifest file that describes all the batches within it and the Batch files themselves.

9 Naming Conventions

Job – A job is identified with a J and a six digit number with zero padding denoted JID (i.e., J000001). As each job is created, this number will increment.

Batch – A batch is identified with a B and a three digit number with zero padding denoted BID.EXT. The extension for a compressed batch is .tar.gz (i.e., B001.tar.gz).

Job Manifest – A Job Manifest file must be named simply JobManifest.xml

Batch Manifest – A Batch Manifest file must be named simply BatchManifest.xml

FITS Files – See 5.1