[illegible]

IMSV0-PD-SD-002 v2.2

EOSDIS

Information Management System

Messages and Development Data Dictionary

V0 and ASTER/ECS Message Passing Protocol Specification

June, 1998

Prepared by

Raytheon STX Corporation
7701 Greenbelt Rd.
Suite 400
Greenbelt, MD 20770

for

EOSDIS IMS
GSFC Code 505

EOSDIS IMS
Message and Development
Data Dictionary

Reviewed by:

[illegible]

Dr. C. Wrandle Barth
IMS Task Leader

Date:

6/23/98

Approved by:

[illegible]

Mr. Ted Johnson
IMS Section Manager

Date:

7/2/98

EOSDIS IMS Message and Development Data Dictionary

Change History

Change #	Changes	Date
1	Changes to ODL search structure INV, DIR, Prod Request, Browse	4/21/94
2	Changes to ODL search structure INV, DIR, Prod Request, Browse	4/22/94
3	Changes to ODL search structure Browse	5/6/94
4	Add Group=Version	5/23/94
5	Add Group = Version to Message Group Characteristics	5/25/94
6	Add Client Version, Billing ID, IMS Staff, Media Format, Protocol Version, and some other misc. changes.	7/25/94
7	Updates to RX and TX Client and Server	11/12/94
8	Delete Browse_Primary_Purpose and Browse_Production_History	12/6/94
9	Add SERVER_VERSION to VERSION Group	5/15/95
10	Add Chunking info, Package info updates, billing enhancements, ECS_AUTHENTICATOR, Interop, spec overhaul	9/18/95
11	Revisions to keywords, groups	9/28/95
12	Revisions to Message Objects, Groups	5/23/96
13	Add SESSION_ID	6/18/96
14	Add Legend and Browse_Header Groups	6/24/96
15	Extensive cleanup; addition of extended search, subsetting and processing options, data URLs, search by granule ids and by path/row, order status and cancel messages, dataset-based ordering, dataset-specific contact addresses, and ASTER/ECS modifications	5/22/97
16	Documentation of VALIDS; changes to data set ordering; change in placement of geographic subset information; integrated browse only specification; data type of WRS_TYPE; clarification on status codes 19 and 29, new status code 30	9/12/97
17	Minor formatting modifications and typos corrected	9/24/97
18	Account-status messages, G-ring coverage, default values for specialized criteria, data center id in inventory and directory searches and in QUITs from data centers	6/17/98

Table of Contents

1. INTRODUCTION

This document defines the message structure to be used in the base protocol transfer between Version 0 Clients (including the WWW/V0 Gateway) and Version 0 Servers (including ECS V0/V1 Gateways which speak the V0 base protocol) and includes information on messages and fields added to support the ASTER/ECS gateway. Some parts of these messages are never generated or are ignored by some of these systems. Some messages, such as those describing order status queries, are being proposed and discussed and may still be subject to change. This document also defines "chunking algorithm" used to segment large inventory results messages into smaller pieces.

One advantage in processing ODL is that any added fields do not affect existing code as long as that code is not looking for those fields. Beyond this base protocol there may be additional messages and additional fields in messages passed by some V0 Clients and V0 Servers. The state machine details how to respond to unexpected requests. Additional fields in base ODL requests may be ignored; indeed, the servers will not even notice them if they do not look for them. Additional fields in replies will always be optional; servers should not need to provide information outside this base protocol.

2. THE V0 PROTOCOL

The basic V0 Protocol consists of a set of requests sent by a client and responses sent by a server. (In some cases, a gateway may act as either client or server or both in exchanging messages. Throughout this document the term "client" should be understood to include gateways acting as clients.) Each connection is initiated by a client to a server socket, and a single ODL message is sent to make a request. The client listens for a response to its request on a socket and processes the result. For many messages, a single exchange of messages completes the transaction and the socket connection is closed. A few, notably inventory requests and integrated browse requests, require an exchange of multiple messages before the transaction is complete.

The basic messages are constructed in Object Description Language (ODL). These messages consist of an internal tree of unordered, labeled nodes. Nodes can be primitives of several types (including strings, integers, real numbers, enumerated symbols and sequences of these) or can be subtrees (called aggregates). Aggregates may be repeated any number of times in the tree. The ODL library provides routines to translate between internal memory representations of these trees and external flat-ASCII representations. The IMS IK-library uses these routines to further provide services for transmitting and receiving trees over sockets. This forms the basis for message communication.

Appendix A lists all fields of the V0 protocol messages and describes the data type and meaning of each. Appendix B lists all aggregates and shows the subfields possible using a BNF-style notation. Because the ODL library provides routines to extract fields of interest, additional fields can be passed without problem. As a result, this document should be viewed as mainly defining the fields that must be present in each message and their meanings. As additional capabilities are added to clients and servers, an attempt has been made to keep all new fields optional so as to support backward compatibility and prevent any requirement that clients and servers be changed in lockstep.

Transaction processing is described by a pair of finite state machines listed in Appendix D. These machines, one for the client and one for the server, define the actions taken as each message is received.

Several new messages were added in version 2.0 of this data dictionary. In addition the source format of the dictionary was changed from a word processing document to a database. This information is also available from the V0 IMS Home Page: http://harp.gsfc.nasa.gov/v0ims/

2.1 What's New

The following items have been added or modified since the previous major release of this document (2.1) in September, 1997.

- A new ACCOUNT_STATUS_REQUEST and ACCOUNT_STATUS_RESULT pair have been added to let ASF and other data centers who maintain billing account allow users to query their balance

- Coverage may now be reported using a G_RING_SPEC to provide a better description of the area covered by a granule; a bounding box for the coverage must still be supplied using a RANGE_SPEC or POLYGON_SPEC to accommodate older clients
- DATA_CENTER_ID has been added as an optional field in both INVENTORY_SEARCH and DIRECTORY_SEARCH since the Web gateway has been including this field for it's own purposes; servers should probably continue to ignore it
- DATA_CENTER_ID, while still technically optional in QUIT messages, must always be provided when the QUIT is sent by a server; it is not included in abort QUITs sent by the client; a note to this effect has been added
- The value ACADEMIC for TYPE has been replaced by K-12 and UNIVERSITY
- The value ORTHOGRAPHIC has been added to MAP_PROJECTION_TYPE
- SPATIAL and TEMPORAL in the valids file has been changed from a single 80-character string to a sequence of 256-character strings; as in all other cases, where a sequence of values is expected a single value is treated as a sequence of one, so this is backward compatible with the previous definition; the content of these fields remains freeform text
- The value **xxx* has been added to PACKAGE_ID
- A new CRITERIA_DEFAULT field has been added to SPECIALIZED_CRITERIA~1
- The following typos were corrected: VALID_ACCOUNT replaces VALID_ACCOUNTS; AUTHENTICATION replaces AUTHENTICATON; PACKAGE_ID is required in LINE_ITEM

The follow items were added or modified to the document in release 2.1 (relative to the previous release, 2.0 in May, 1997).

- The definition of the valids submission file has been added to the dictionary, along with text describing this file
- The definition of fields related to ordering by dataset have been modified in accordance with experience in implementing this feature, specifically:
  - PACKAGE_ID can be specified in a DATASET~1 or DATASET~2 to name packaging information to be associated with a dataset as a whole and to indicate it may be ordered as a whole
  - (PACKAGE)* may be specified in a DIRECTORY_RESULT or DATASET~2 to pass packaging information back with directory searches to allow ordering datasets as a whole
  - A new group, DATASET_ORDER_OPTIONS, has been added to the PACKAGE to allow specification of SPECIALIZED_CRITERIA~1 for dataset ordering in the same way SUBSET_OPTIONS and ORDER_OPTIONS are; DATASET_ORDER_SPATIAL and _TEMPORAL have been removed from DATASET~2 since they are redundant and not parallel to other option specifications
  - A new group, DATASET_ORDER_SPEC, has been added to the LINE_ITEM group to indicate the DATASET_ORDER_OPTIONS selected parallel to the SUBSET_SPEC and ORDER_SPEC, replacing the RANGE_LOC, START_DATE, and STOP_DATE fields previously provided
- The specification of geographic valued SUBSET_SPEC or ORDER_SPEC through a POINT_LOC, POLYGON_LOC~1, or RANGE_LOC has been moved inside the SPECIALIZED_CRITERIA~2 group to provide consistency with other specifications and as well as to make it available for EXTENDED_SEARCH and DATASET_ORDER_SPEC

- The new field INTEGRATED_BROWSE_ONLY was added to GRANULEs to be able to specify that integrated browse can be requested but not ftp browse; this was made a new field rather than a new value of the existing BROWSE_TYPE to improve backward compatibility
- The definition of WRS_TYPE was modified to match the implementation
- Status codes 19 and 29 have slightly changed, more precise meanings; status code 30 has been added

The follow items were added or modified to the document in release 2.0 (relative to the previous release, 1.0.18).

- Extensive cleanup; there were numerous errors in the parent/children listings, in optional fields that were not marked as being optional, and in fields that were used differently in various environments; to point out these essentially different definitions in this latter case, a new convention has been introduced using a tilde and digit following definitions that differ but use the same keyword; in addition, the data dictionary has been more rigorously defined to allow the creation of a protocol verifier tool for use in checking ODL messages being exchanged between ASTER and ECS
- Elimination of some fields that were never implemented, including XHAIRS and PACKAGE_CONTACT_ADDRESS; also elimination of ORG_CENTER in DIRECTORY_RESULT as a meaningful field and USER_AFFILIATION as a required field in BROWSE_REQUESTs
- ASTER/ECS additions; those additions that have been made for ASTER/ECS only and are not becoming a part of the basic V0 protocol have been identified in Appendix A in the notes for each field; these include:
  - Addition of optional fields XAR_ID and CLOUD_COVERAGE to INVENTORY_REQUEST
  - Addition of optional fields XAR_ID, SCENE_CLOUD_COVERAGE, and QUADRANT_CLOUD_COVERAGE to GRANULE in INVENTORY_RESULT
  - Mechanism to return multiple browse files in response to a single integrated browse request; includes an optional LAST_BROWSE=0 in INTEGRATED_BROWSE_RESULT message other than the last, ACKNOWLEDGE by the client after each image is received (other than the last), and looping in the state machine to receive additional INTEGRATED_BROWSE_RESULT messages and images
  - Change in structure of PRODUCT_REQUEST replacing LINE_ITEM and its substructure with a new aggregate called MEDIA, that insures grouping by media type and format; also use of TYPE_ID and FORMAT_ID in place of MEDIA_TYPE~2 and MEDIA_FORMAT~2 keywords, and use of INITIATOR_REQUEST_ID in place of REQUEST_ID; additional fields under MEDIA allow specification of SENSOR_TYPE, PRODUCT_TYPE, and PRODUCT_GENERATION parameters (similar to, but less flexible than, the ORDER_SPEC capabilities being added for ASF and others)
  - New PRICE_ESTIMATE_REQUEST and _RESULT added to get price estimate within the order process
  - New variation of dataset information DATASET~3 returned in DIRECTORY_RESULT that gives GCMD-like information directly without requiring separate query to GCMD

- ASTER/ECS additions also being adopted for V0 use
  - New PRODUCT_STATUS_REQUEST and PRODUCT_STATUS_INFO messages to query status of order; there are a few differences in some fields between the ASTER/ECS version and the V0 version
  - New PRODUCT_CANCEL_REQUEST and PRODUCT_CANCEL_RESULT message to cancel order or suborder; again there are a few differences
- V0 extensions that came largely out of the work of Helen Conover with the V0 DAACs and the international community, including:
  - Extended search capability: allowing users to specify extended searches using dataset-specific criteria; includes definition of new support file information containing criteria and dependencies with other valids, new optional EXTENDED_SEARCH group in INVENTORY_SEARCHES, and new optional EXTENDED_CRITERIA_USED and SPECIALIZED_RESULTS groups within INVENTORY_RESULTs returned by DAACs choosing to implement these searches
  - Search by path/row: allowing client to return Landsat path/row coordinates selected by the user in addition to existing geographic location information; knowledgeable servers can use this to return granules specific to this measuring system; also allows coverage in GRANULE to be returned by path/row in addition to other geographic terms; in all cases clients and servers not aware of this information can ignore it and use conventional specifications
  - Search by granule id: allowing user to specify one or more strings of granule ids (perhaps with wildcards embedded) for retrieval from a specific dataset; new optional field in INVENTORY_SEARCH of GRANULE_ID_REQ to specify search string; geographic information is not required for this search and servers not recognizing GRANULE_ID_REQ field will simply respond often with a message about failure to specify geographic area of search; this addition has made the geographic specification, previously required, now optional; source, sensor, campaign, processing level, and parameter are not available with search by granule id; dataset name is required in this case
  - Granule subsetting during order: new optional SUBSET_OPTIONS group returned in PACKAGE information of INVENTORY_RESULT names SPECIALIZED_CRITERIA that may be used to specify subsetting during ordering; no change required by DAACs not wishing to implement this; new optional SUBSET_SPEC in LINE_ITEM returns values selected by users
  - Product generation specification: new optional ORDER_OPTIONS group returned in PACKAGE information of INVENTORY_RESULT names SPECIALIZED_CRITERIA that may be used to specify product generation parameters to be used, analogous to the SUBSET_OPTIONS above; new optional ORDER_SPEC in LINE_ITEM returns values selected by users
  - Dataset URLs: optional groups DATA_URL, DATASET_HOME_PAGE, SPECIALIZED_SEARCH_URL, BROWSE_URL, and MISC_UL have been added to a number of areas to allow servers to return pointers to additional capabilities to clients
  - Dataset-based ordering: several new optional fields have been added to DATASET~2 to allow the possibility of ordering directly from a directory search; this is a new area being explored by the V0 team
  - A new value of POLE_INCLUDED=B was added for datasets covering both poles

2.2 Basic Protocol

The basic protocol consists of a set of request and response messages implemented between V0 clients (including the Web gateway) and V0 servers (including those of various international partners), and include certain ancillary messages used to support these (such as ACKNOWLEDGE and QUIT). The structure of each message is show in Appendix B. The definitions of the individual fields are shown in Appendix A. The state machines that control the passing of the messages is shown in Appendix D. Every message is converted from an ODL memory tree to a flat text file (called an ODL label), which is then sent on a Unix socket, preceded by a four-byte integer size field (transmitted in network order) that gives the length of the label. When received the label is converted back to a memory tree. The low-level routines of the V0 IMS IK library that transmit and receive socket information also handle conversion between trees and labels, transmission and receipt of the size field, and addition and updating of the MONITOR and VERSION groups.

Note that the basic protocol does not include support for guide document searching. These searches are performed using WAIS queries against standard WAIS servers, and http protocol for following internal hypertext linkage. No ODL messages are used to support this.

2.2.1 Inventory Searching

An inventory search is initiated by a client sending an INVENTORY_REQUEST message. Normally this includes at a minimum a source, sensor, or parameter selection and a geographic specification. Servers are expected to return information about granules that satisfy at least one of the values in each of the fields requested (i.e., the values within each field are ORed for the search, and the fields are ANDed together. In addition to the traditional fields (source, sensor, parameter, processing level, data center, dataset, campaign, geographic and temporal limits, browse-only, and day/night flag), new fields have recently been defined. For ASTER/ECS, CLOUD_COVERAGE and XAR_ID are considered to be special search criteria that may be specified and will be ANDed with other criteria. For V0, the new EXTENDED_SEARCH group allows dataset-specific search criteria to be added; this will handle not only items like cloud cover but provide an extensible mechanism whereby new criteria can be added by the science team without requiring further changes to the protocol.

An alternative search is provided if the GRANULE_ID_REQ field is present. In this case, specific granule identifiers (possibly including wildcard characters) are being requested within specific DATASET_ID values which must accompany them. Source, sensor, campaign, parameter, and processing level may not be specified with a granule id search.

Servers respond with a sequence of zero or more INVENTORY_RESPONSE messages, each of which are acknowledged by the client, followed by a QUIT message (to indicate the end of the transaction). The group of INVENTORY_RESPONSE messages act together to form a single logical INVENTORY_RESPONSE, but are segmented into individual messages, or chunks, to make communications more manageable. A client can also return an ABORT message (actually a form of QUIT) at any time to request premature termination of the search. The rules for segmenting the INVENTORY_RESPONSE into chunks are discussed in section 3.

Included in the INVENTORY_RESPONSE is metadata for individual granules matching the inventory request. Information is also returned indicating which granules have browse products available. PACKAGE information is also returned that provide the information to allow the user to order packages containing the granules.

2.2.2 Directory Searching

A directory search is initiated by a client sending a DIRECTORY_REQUEST message. Servers respond with a DIRECTORY_RESPONSE message whose primary function is to provide the Global Change Master Directory DIF identifier for each matching dataset. Since these identifiers are unique, the formerly optional ORG_CENTER has been dropped and will be ignored if transmitted. Newly added to the DIRECTORY_RESPONSE are optional fields to allow data centers to provide information to allow clients to do dataset-based ordering.

2.2.3 Browse Requests

Using information returned in an inventory search, a user can request browse products either to be staged for ftp pickup or to be returned directly through the V0 protocol. The client sends a BROWSE_REQUEST message which includes a BROWSE_TYPE field to indicate whether an ftp or integrated transfer is desired. The server responds with either an FTP_BROWSE_RESULT message to acknowledge the receipt of the request (the details of the ftp pickup are transmitted in a subsequent email message to the requester) or with a sequence of browse files each prefixed by an INTEGRATED_BROWSE_RESULT message which contains the size of the file that follows it. V0 systems are normally limited to a single integrated browse file per request. The ability to handle multiple files per request was added for ASTER/ECS. A special flag LAST_BROWSE=0 is returned in all but the last INTEGRATED_BROWSE_RESULT which will prompt knowledgeable clients to return an ACKNOWLEDGE and loop for the next file. A client not programmed to handle multiple files will overlook the LAST_BROWSE=0 flag and terminate the connection after receiving the first file, which will result in the server shutting its communication with that client due to a unexpectedly closed connection. If a client that can handle multiple files talks to a server that cannot, the single file returned will not have the LAST_BROWSE=0 flag and will be treated as simply returning only a single file.

2.2.4 Product Request

Using information returned in an inventory search, a user can order product packages from the data center. A PRODUCT_REQUEST message is sent by the client to a data center which describes one or more LINE_ITEMs to be ordered. (ASTER/ECS require orders to be pre-grouped by media, so the MEDIA group replaces the V0 LINE_ITEM group.) Once the PRODUCT_REQUEST is received by a server, it returns a PRODUCT_RESULT to acknowledge the request. Tracking information is returned in the result, since there is no guarantee of acceptance of the request by the result message.

2.2.5 Account Status

While most archives do not support accounting or billing, a few do. With version 2.2 of this Data Dictionary, a new request has been added to query the status of an a user's accounts. The client sends the same CONTACT_ADDRESS information in an ACCOUNT_STATUS_REQUEST that would be sent with a product request and receives back an ACCOUNT_STATUS_RESULT listing all of the user's VALID_ACCOUNTs. A new ACCOUNT_COMMENT field has been added to allow arbitrary text information to be returned providing archive-specific accounting information.

2.3 ASTER/ECS Extensions

With the exception of a few areas noted above, the ASTER/ECS uses the same protocol to accomplish the requests of the basic protocol. However a few additional messages were added for ASTER/ECS use. The messages exchanged between ASTER GDS and the ASTER Gateway provided by ECS are sometimes referred to as the V0' protocol. Some of these are also being adopted as V0 extensions as well.

2.3.1 Directory Searching

In order to avoid the need for ASTER clients to search the GCMD directly the way V0 clients do, directory information has been added to the DIRECTORY_RESULT message returned by the ASTER Gateway. These fields form the group described as DATASET~3 in the appendices.

2.3.2 Product Status

A product status request allows a client to request status on orders using request ids. In ASTER/ECS, this is the INITIATOR_REQUEST_ID assigned by the initiating side of the order. For V0 adoption, this is the REQUEST_ID~2 which may be the original REQUEST_ID~1 assigned by the client, the DAAC_ORDER_ID assigned by the data center, or a combination. Not all data centers are expected to necessarily handle all forms. In fact, the DAAC_ORDER_ID was added specifically for data centers who had no way to track orders by a client-selected id. The server returns a PRODUCT_STATUS_INFO message which may break the order into subrequests, each with their own individual status.

2.3.3 Product Cancel

A product cancel request allows a client to request cancellation of a complete order or of subrequests within an order. The operational scenario for canceling a subrequest presumes the user issues a product status request first to obtain subrequest ids for use in the cancel request. The PRODUCT_CANCEL_REQUEST can then be sent to the server for the full order or for one or more subrequests. The server returns a PRODUCT_CANCEL_RESULT to indicate its handling of the request. V0 is considering adoption of this message.

2.3.4 Price Estimate

The V0 protocol contains insufficient information in the PACKAGE group for ASTER/ECS to calculate a price estimate for the user. To provide greater flexibility in estimating the price, a new message has been added that can allow the estimate to take into account product generation parameters. The PRICE_ESTIMATE_REQUEST message is always issued before a product request in the ASTER/ECS environment by the client and the server returns a PRICE_ESTIMATE_RESULT.

3. CHUNKING PROTOCOL

When the Inventory Results generated from a user query are large, an Inventory Results message can be broken up into "chunks" according to a set of rules. Chunking helps breakup large Inventory Results into smaller but complete trees. The chunks are composed of basic types of information; Inventory Result Prefix, Dataset group, and Granule Group. Package Information can be integrated into the tree according to three options:

Option 1. - Adding All Package Groups in front of the First Dataset Group
Option 2. - Adding Relevant Package Groups in front of each Dataset Group
Option 3. - Adding Relevant Package Groups in each Dataset Group

The following example illustrates the structure, guidelines, and options for placing Package Information for chunking:

INVENTORY RESULT PREFIX:
Info: (Message_Id, Data_Center, Status_Code, Status_Code_Comment, Unmapped_Field)
Rule: (Required for each tree/chunk)

Option 1 for Package Information (0 - many per chunk)
Option 2 for Package Information (0 - many per chunk)

DATASET GROUP
Info: (Metadata within the Dataset group)
Rule: (0 - many [current restriction is: 1 - many]; avoid repeating in other chunks)

Option 3 for Package Information (0 - many per chunk)

GRANULE GROUP
Info: (Metadata within the Granule group)
Rule: (0 - many per chunk)

The current implementation requires each chunk contain at least Inventory Result Prefix information and Dataset group metadata. When this restriction is removed, then a 0 - or - more option will permit the following combinations of information for chunks:

a. Chunk - Inv Result Prefix + Package Information
b. Chunk - Inv Result Prefix + Package Information + Dataset metadata
c. Chunk - Inv Result Prefix + Dataset metadata + Granules
d. Chunk - Inv Result Prefix + Dataset metadata + Package Information + Granules

Chunking can specify the total number of granules returned in an Inventory Results message. The size of each chunk need not be uniform in size although a past guideline constraint for a granule-per-chunk cap of 51, yielding a chunk size of 64Kbytes, is useful but not mandatory. Figure 1 depicts an example of the possible ways information in a 'chunked' Inventory Result message can be organized. Note that the number of granules per dataset may or may not fit in one chunk. The average size allowed for a chunk helps decide what combination of information will be fit into it. This is affected by the different options selected for sending package information.

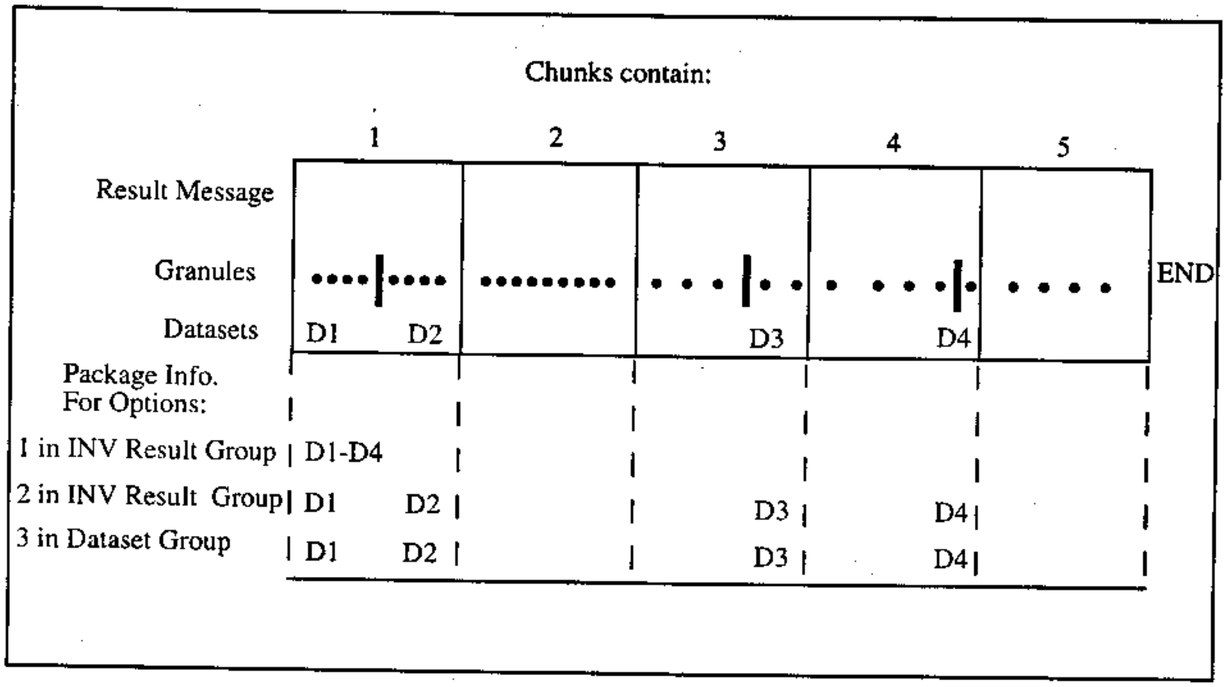

Figure 1: Illustration of chunking an Inventory Results message

3.1 Rules and Guidelines

Size of each chunk need not be uniform but should be moderate in size (64kbytes).

The size of the package information for granules in a particular Inventory Results message can be either consolidated in the first chunk (Option 1 if it fits within the chunk size cap) or distributed in other chunks according to Options 2 and 3.

Avoid repeating the Dataset metadata and package information once provided in an appropriate chunk.

Granules are added to a dataset group until the granules per chunk limit is reached. Remaining granules can be put into the next chunk(s).

Use the NUMBER_OF_GRANULE_HITS field to store the total granule count for the result message following the last granule of each dataset in the sequence.

An Inventory Result chunk can have several dataset groups or
granules from dataset can be spread across several Inventory Result chunks.

Do not break ODL trees across groups (any chunk is a complete tree).

Chunks come in a sequence.

Each Result message is expected to have the Message_ID and Monitor group added.

Dataset metadata is included only before the first granule.

A chunk can contain more than one dataset and their granules.

4. VALIDS SUBMISSION

The V0 clients present users with lists of valid values for various fields from which to construct their searches. In addition to knowing these values, the clients know dependencies among them. This allows them to narrow selections for various fields based on values entered in earlier fields. It also allows the client to limit the data centers to which it sends the request even when the user does not make such limitations himself.

The basis for these valid dependencies is in information provided by the archives. Whenever an archive wants to add new datasets or otherwise change its valids, it is currently required to send to the V0 Science and Operations team a file containing an ODL tree (label) describing the valids for its entire holdings. This is combined with the most recently submitted tree from all other archives to form a set of support files used by the clients to handle query construction. While discussions are ongoing about changing this from a batch system to some form of transactional database, the current document describes the format of the ODL tree as it is currently implemented.

The submitted file contains what amounts to a single ODL group called VALIDS followed by a single line containing "END". The VALIDS group contains a DATA_CENTER_ID field naming the archive and one or more DATASET~4 groups describing the datasets available from that archive. In practice, it is preferable to submit a file which actually leaves off the surrounding GROUP=VALIDS … END_GROUP = VALIDS lines and just contains the DATA_CENTER_ID field, one or more DATASET~4 groups, and the final END line.

The DATASET~4 includes entries for SOURCE, SENSOR, PARAMETER, and a DEPENDENCY group, among others. The DEPENDENCY group can also have SOURCE, SENSOR, and PARAMETER entries, and can be repeated. This allows a data center to simply list the sources, sensors, and parameters that are available within a dataset, or to group them into dependency groups that more completely describe the connections between valids. For example,

```
GROUP = DATASET
        DATASET_ID = "GPCC GLOBAL PRECIPITATION"
                  ;
        SOURCE = ("NOAA", "GMS", "METEOSAT", "GOES", "DMSP")
        SENSOR = ("SSM/I", "IR")
        PARAMETER = ("PRECIPITATION")
END_GROUP = DATASET
```

indicates that this dataset includes sensors SSM/I and IR which measure precipitation from the listed sources. However,

```
GROUP = DATASET
        DATASET_ID = "GPCC GLOBAL PRECIPITATION"
                  ;
        PARAMETER = ("PRECIPITATION")
        GROUP = DEPENDENCY
                SOURCE = ("NOAA", "GMS", "METEOSAT", "GOES")
                SENSOR = ("SSM/I")
        END_GROUP = DEPENDENCY
        GROUP = DEPENDENCY
                SOURCE = ("DMSP")
                SENSOR = ("IR")
        END_GROUP = DEPENDENCY
END_GROUP = DATASET
```

further indicates the IR sensor is used only for the DMSP data. If a user selects IR and then looks at compatible sources, he will see only DMSP if the valids have been submitted in the latter way; in the former, he would see all five sensors.

Note that PARAMETERs may also be specified within the DEPENDENCY groups. Each of the three fields may be specified at either the DATASET~4 level directly or in all DEPENDENCY groups within the DATASET~4; none may be specified at both levels.

The ingest routine compares valid values to a master list maintained by the Science and Operations staff. This comparison is case blind and the case shifting provided by the master list is used in the final valids, so the case in the valids file is not important. Fields (like SENSOR) that are supposed to be sequence strings may be given as scalars (i.e., the parentheses may be omitted) if only a single value is present: SENSOR = "IR".