

160-WP-002-001

# QA Metadata Update Tool for the ECS Project

White Paper

April 1998

Prepared Under Contract NAS5-60000

## RESPONSIBLE ENGINEER

<u>Sushma Singhal /s/</u>	<u>4/22/98</u>
Sushma Singhal, Systems Engineer EOSDIS Core System Project	Date

## SUBMITTED BY

<u>R. J. Plante /s/</u>	<u>4/22/98</u>
Robert Plante, Science Department Manager EOSDIS Core System Project	Date

Raytheon Information Systems Company  
Upper Marlboro, Maryland

This page intentionally left blank.

# Abstract

---

The ECS provided Metadata Quality Assurance (QA) functionality supports the update of Science and Operational QA metadata for data granules from the perspective of defining quality control procedures "after" the granule has been created. The QA Metadata Update Tool's (QA MUT) purpose is to enable SCF and DAAC QA experts to modify ScienceQualityFlag and OperationalQualityFlag attributes of core metadata for multiple granules at a time (vs. One granule at a time). The tool consists of two major components: The user component and the DAAC component. The user component submits the update requests to the DAAC operator's component via e-mail. The DAAC operator's component of the tool transmits the update requests to the Science Data Server, format an e-mail message to notify the requester about the status of the update request, and transmits the status message back to the requester. This paper describes the current high level design of the MUT (version 1).

**Keywords:** DAAC Operator, JEST, QA Metadata Update, OperationalQualityFlag, SCF, Science Data Server, ScienceQualityFlag, SDSRV

This page intentionally left blank.

# Contents

---

## Abstract

### 1. Introduction

1.1 Purpose.....	1-1
1.2 Organization.....	1-1
1.3 Review and Approval.....	1-2
1.4 Acknowledgments.....	1-2

### 2. Context in the Overall ECS System

2.1 QA Metadata Update .....	2-1
2.2 Basic Capability of the QA Metadata Update tool .....	2-1
2.2.1 Capabilities for the End User.....	2-1
2.2.2 Capabilities at the DAAC .....	2-2
2.3 High Level Science QA Metadata Update Scenario.....	2-2

### 3. Design Overview

3.1 Capabilities.....	3-1
3.2 QA Metadata Update User Interface.....	3-3
3.2.1 Security and User Profile Variables.....	3-3
3.2.2 Search Query Through JEST .....	3-4
3.2.3 QA Metadata Updates.....	3-5
3.2.4 QA Updates Transmission .....	3-9
3.3 DAAC interface .....	3-9
3.3.1 Security on the DAAC operator tool.....	3-10
3.3.2 User Notification.....	3-10
3.3.3 Update History .....	3-11

## List of Figures

3-1. QA Metadata Update Tool Within ECS.....	3-1
3.2.1. User Authorization for QA Update.....	3-3
3.2.2A. The JEST Search Query Screen Layout.....	3-4
3.2.2B. The JEST Query Results Screen Layout.....	3-5
3.2.3A. The QAMUT User Screen Layout.....	3-6
3.2.3B. The QAMUT User Screen Layout.....	3-7
3.2.3C. The modified values to QA Metadata.....	3-8
3-4. The Update Message.....	3-10

## List of Tables

2-1. Scenario 1 - Routine or Troubleshooting QA Updates.....	2-3
--	-----

## Appendix A. QA Related Core Metadata from the B.0 Data Model

## Appendix B. Format of the Message Returned to the Requester

## Abbreviations and Acronyms

# 1. Introduction

---

## 1.1 Purpose

This paper describes the capabilities and design of the Quality Assurance (QA) Metadata Update Tool (QAMUT). The QA Metadata Update Tool's purpose is to enable both Science Computing Facility (SCF) and Distributed Active Archive Center (DAAC) QA experts to modify values of their respective quality flags (i.e., ScienceQualityFlag and OperationalQualityFlag) on core metadata for multiple granules at a time in a batch mode. To support these functionality, the tool must:

- Restrict update privileges to only authorized users
- Accept a list of multiple granules to update (batch update)
- Ensure that metadata updates are performed using valid values
- Maintain a log of metadata changes and the source of the changes

These capabilities, when combined with DAAC and SCF operational procedures, will enable authorized users to quickly and easily modify the QA metadata for which they are responsible while maintaining a system of checks and balances to reduce the risk of malicious or accidental corruption of EOSDIS products.

This paper describes the ECS provided QA functionality, QA update scenario, functional and operational capabilities of the tool (version 1.0), and some design issues to be resolved.

## 1.2 Organization

This paper is organized as follows:

Section 1 - States the purpose of the QA Metadata Update Tool.

Section 2 - Provides the context of the QA Metadata update in the overall ECS system

Section 3 - Includes the current high level design of the QA Metadata Update Tool

Appendix A - Lists the QA Metadata attributes

Appendix B. Provides Format of the Message Returned to the Requester

Abbreviations and Acronyms

This is an informal technical paper not intended for formal review or government approval. The formal review comments are not required but can be provided. Questions regarding technical information contained within this Paper should be addressed to the following ECS contact:

- Sushma Singhal, RSTX, (301) 883-4095, [ssinghal@eos.hitc.com](mailto:ssinghal@eos.hitc.com)

### **1.3 Review and Approval**

Questions concerning distribution or control of this document should be addressed to:

Data Management Office  
The ECS Project Office  
Raytheon Information Systems Company  
1616 McCormick Drive  
Upper Marlboro, MD 20774-5372

### **1.4 Acknowledgments**

The following people are acknowledged for having contributed to the writing of this document:

- Cheryl Croft, RSC
- Rick Hatfield, RSC
- William Sarver, RSC

## 2. Context in the Overall ECS System

---

The QA Metadata Update Tool's purpose is to enable Science Computing Facility (SCF) and Distributed Active Archive Center (DAAC) QA experts to modify ScienceQualityFlag and OperationalQualityFlag<sup>1</sup> attributes of core metadata for multiple granules in a batch mode at a time (vs. one granule at a time). The tool consists of two major components: The SCF component and the DAAC component. The DAAC component generates a status message and transmits the status to the requester.

### 2.1 QA Metadata Update

The ECS provided QA functionality supports the update of Science and Operational QA metadata for data granules from the perspective of defining quality control procedures "after" the granule has been created. Furthermore, it is quite possible that during the first six months after launch instrument team and DAAC personnel will spend considerable time in performing error handling analysis and refining the science algorithms within the software, and may have little time for quality assessment of their data products. Therefore, one of the goals of the QA Metadata update tool is to help ensure that each SCF will be able to perform QA of their products shortly after launch by making this process easy and efficient.

### 2.2 Basic Capability of the QA Metadata Update tool

The basic capabilities supported by the QA Metadata Update Tool (MUT) are described below. The discussion that follows focuses on SCF use of the tool; DAAC operational use is equivalent but changes OperationalQualityFlag.

#### 2.2.1 Capabilities for the End User

- The Java Earth Search Tool (JEST) will be used to prepare the list of granules that are to have metadata values updated. The QA Metadata Update tool will accept the list of data granules resulting from the search criteria entered from the JEST client. The list of granules displayed will include granules that are to have science QA values or operational QA values applied. Full content of the granule level information to be displayed is user-defined. In addition to the automatically provided Universal Reference (UR), we recommend displaying additional descriptive metadata that would help identify each UR.
- The QA Metadata Update Interface provides a graphical user interface enabling authorized users to edit the quality flags of the granules provided by JEST.
- The QAMUT provides the necessary means (e.g., basic editing capability) to associate valid ScienceQualityFlag or OperationalQualityFlag update values to the granules in the list. The SCF QA Metadata Update Interface will allow both changes

---

<sup>1</sup> See Appendix A for definition of quality-related attributes.

to an individual granule as well as to the selections or groups of granules from within the displayed list.

- The update requests consisting of a list of URs, MeasuredParameters, and QualityFlags including FlagExplanation values will be formatted into a standard format, referred to here as the QA Metadata Update Message.
- The resulting formatted request will be sent via electronic mail to a predefined email address at the appropriate DAAC.

### **2.2.2 Capabilities at the DAAC**

- The QA update requests will be received by the DAAC personnel responsible for performing the updates to the science database. The physical presence of a DAAC representative is required to meet security requirements. A non-GUI based script is run by the DAAC operations that updates the metadata values in the Science Data Server with the values contained in the update message. Any required DAAC operator inspection will be conducted using conventional UNIX tools or email tool.
- An update history will be maintained at the DAAC for the modified QA values in the form of saved QA update messages.
- An e-mail notification regarding the updates performed will be sent to the requester at the address provided in the initial message.
- The operation staff at the DAAC will also have access to the QA Metadata Update Tool to perform updates to the OperationalQualityFlag, creating a QA Metadata Update Message in the same manner as the ITs performing Science QA Metadata updates at the SCF. The Operations QA update will be applied to the inventory using the same scripts as for the Science QA update. Configuration parameters used with the tool will indicate the QA attributes (Science or Operational) that are to be updated.

## **2.3 High Level Science QA Metadata Update Scenario**

The scenario described here illustrates use of the QA Metadata Update tool. The scenario is based on the following assumptions:

1. The IT will routinely evaluate a sample of the science data (e.g., 10% of the data) as part of the QA program for the science data. The size of the sample and the selection criteria will be specified by the IT.
2. Certain automated checks will be performed by the PGEs resulting in setting of the Automatic Quality Flags. Results from this action will often require manual inspection of questionable products by the IT.

3. Science data QA investigations and resulting metadata updates can also be expected to occur as a result of spot checks, coincidental discovery of QA problems by ITs, and by notification from other science data users.
4. The ITs are responsible for the generation of the QA update information, as a part of their QA program, according to the plans and schedules established by the ITs.
5. DAACs will be responsible for updating this QA information into the Science Data Server (SDSRV) as it arrives after suitable inspection of the QA metadata values for reasonableness. The DAACs participation is required for security purposes. The current design of the QA Metadata Update tool (version 1) does not incorporate security features of Access Control List (ACL) and Distributed Computing Environment (DCE). Therefore, DAAC's participation is required in updating the databases for security and to maintain the integrity of the ECS databases.
6. If an undo operation is needed either as the result of a user's error in QA value entry or because of the changed assessment of the data quality, the IT will prepare a new QA Update Message containing the revisions to the fields that need to be changed.
7. The history of Science QA updates will be maintained for a predefined period only at DAACs. Following the QA updates at the DAAC, an e-mail message will be automatically generated and sent to the originating user notifying the status of update request.

The Table 2-1 lists the steps included in the scenario.

**Table 2-1. Scenario 1 - Routine or Troubleshooting QA Updates (1 of 2)**

	Step	Comment
1.	Initial condition: A Product Generation Executive (PGE) is running.	--
2.	PGE produces (directly or indirectly): <ul style="list-style-type: none"> <li>• Data Products</li> <li>• Metadata, including Core QA-related metadata (QAStats &amp; AutomaticQualityFlag) &amp; PSA QA metadata</li> <li>• "Supporting Information" files</li> <li>• Production History, incl. User Log, related info</li> </ul> Note: Success or failure of the run is determined by the PGE and this assessment is provided to ECS at PGE exit.	All supported by standard ECS services
3.	PGE completes, data is destaged, metadata stored to DSS. <ol style="list-style-type: none"> <li>a. AutomaticQualityFlag indicates acceptable results based on PGE assessment.</li> <li>b. AutomaticQualityFlag indicates data anomaly based on PGE assessment.</li> </ol>	Automatic QA Set Core metadata update PSA metadata update

**Table 2-1. Scenario 1 - Routine or Troubleshooting QA Updates (2 of 2)**

	<b>Step</b>	<b>Comment</b>
4.	Based on IT-defined criteria, the granule for this PGE is identified for “representative sample” examination. Data, metadata, supporting information, and production histories are retrieved to the SCF.	Options for retrieval specification: a. Qualified subscriptions (Supported in Release 2) –subscriptions with qualifiers against core and PSA metadata. Can be used to stage granules to SCF for evaluation. ITs may need to manually retrieve supporting information. b. Use of Client – Via the JEST client, the user explicitly selects and retrieves the granules needed for evaluation and the supporting data. Client can save the retrieval criteria for subsequent editing and reuse.
5.	IT personnel evaluates the data products and supporting information at the SCF.	Tools for this analysis and display are IT provided and specific to the data products.
6.	Using the QAMUT interface, IT personnel construct the Science QA metadata update message in the form of a list of metadata (ShortName, start time, stop time, etc.) for each measured parameter within the granule with science QA metadata update values included. These update values can be both for those explicitly evaluated and those whose status is inferred from the granules that were explicitly evaluated.	Using the Client, the IT produces a results list in standard format (TBD content) for those granules whose QA values are to be updated. Results list is imported to the Science QA Metadata Update tool. The IT updates the entry/record for each granule with appropriate QA value. The QA Metadata Update tool outputs the resulting message in a standard format, ready for email.
7.	The QAMUT automatically transmits the QA metadata update message via email to the predefined email address.	The user profile files contains the electronic addresses to which the email is to be sent.
8.	DAAC operations personnel run science QA metadata update application using Science QA metadata update message as input.	The QA update script at the DAAC will parse the email message, extracting QA metadata values and updating the metadata accordingly via the SDSRV APIs.
9.	Upon completion, DAAC operations tool sends an e-mail confirmation of updates to SCF user.	Needed to confirm the closure of the metadata update operation.

Note that coincidentally discovered data quality problems can lead to QA metadata update changes in the same manner as scenario 1, i.e., interactive selection of granules to be reviewed, search results list used as a vehicle for transfer of QA metadata updates.

The scenario for the update of the operational quality flag is fundamentally the same. The operations QA staff select certain granules of data for evaluation at the DAAC. The QA staff members will use EoS view, the data visualization tool provided for this purpose, to inspect the granules for anomalies. This inspection is not intended as a detailed technical inspection for science quality but rather a quick inspection for obvious problems that might have arisen during processing. The EoS view tool is suitable for this type of inspection. The EoS view tool is applicable only on EOS-HDF format granules, which is the standard format for EOS data. Once the granule has been assessed, the metadata update can proceed as described above.

### 3. Design Overview

This section describes the capabilities of the QA Metadata Update Tool to be provided for B.0 support and the high level design for the implementation.

#### 3.1 Capabilities

The Figure 3-1 describes the QA MUT's components' interaction and a chronologically ordered scenario. Specific actions at each step are described below.

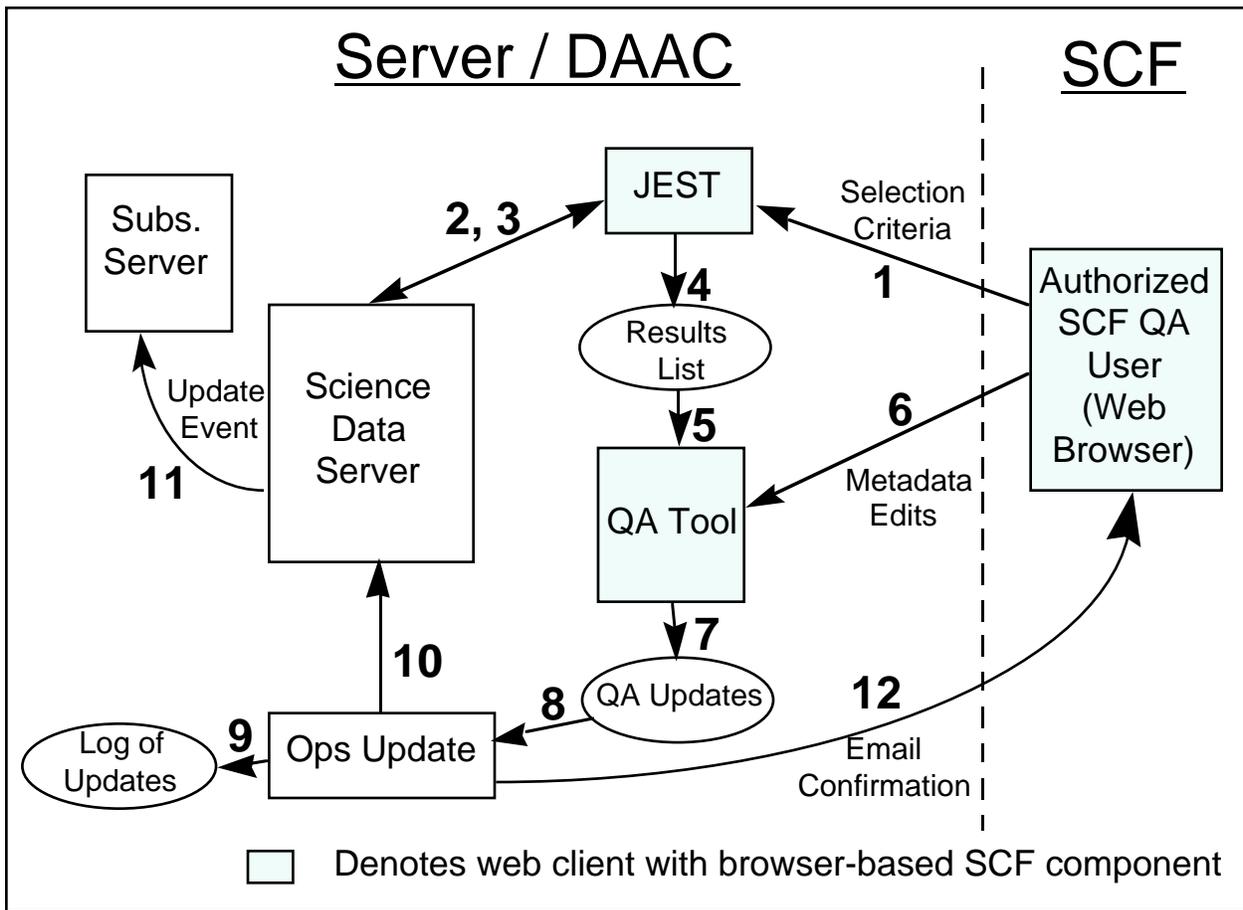


Figure 3-1. QA Metadata Update Tool Within ECS

<b><u>Step</u></b>	<b><u>Action</u></b>
1.	SCF user uses the Java Earth Science Tool (JEST) client to formulate a search based on a user-defined combination of core and product specific metadata.
2.	JEST sends search criteria (stored within JEST's project and search folder metaphors) to Science Data Server to locate matching granules.
3.	In response to the query, the Science Data Server returns a list of granules, with Universal Reference (UR) and any other distinguishing metadata the user specified in the search criteria.
4.	JEST stores the search results on disk available to the web server from which it runs.
5.	JEST will provide an "QA UPDATE" button on the inspect results screen. This button will only be displayed for users with authorization to perform QA metadata update. Using a click-on-operation on the "QA UPDATE" button, the web-based QA Metadata Update tool is launched. Note that the authorized user must log into the appropriate DAAC to perform this operation. If s/he is authorized to update metadata for granules that are saved at different DAACs, s/he must access each DAAC separately through JEST.
6.	The user selects a valid QualityFlag value for each granule in the list, or selects multiple granules and updates all of the selected granule with a single valid value.
7.	Upon selection of the Update option, the tool saves the QA updates request and username on the disk available to the web server from which it runs.
8.	The list of QA metadata updates are formatted in an e-mail message and forwarded to the DAAC operator.
9.	The DAAC operator reviews (using UNIX tools) the update list for its compliance with DAAC/SCF-agreed guidelines. The review criteria will be as defined by operations. The DAAC operator runs the scripts to commit the updates. The update list and SCF source identification are saved to a QA log file.
10.	The updates are passed to the Science Data Server via the DAAC Operator tool. The Science Data Server's inventory metadata are updated for the granule(s) and MeasuredParameter(s) specified.
11.	The Science Data Server sends a metadata update event for each affected granule to the Subscription Server. The subscription server processes all requests for notification of these events.
12.	The DAAC Operator update script formats an email to the SCF user confirming the metadata update.

The design of each component of the QAMUT is described below.

## 3.2 QA Metadata Update User Interface

The QAMUT interface will, like JEST, be web based Graphical User Interface. This is both for ease of use and to simplify the interfaces implementation. The goals of SCF component (i.e., component that is used by people who are at SCFs) of the QAMUT is to accept a list of granules resulting from a query entered by the user at the front end JEST client and enable user to modify values of (ScienceQualityFlag, ScienceQualityFlagExplanation) or (OperationalQualityFlag, OperationalQualityFlagExplanation) pairs in a valid format for the DAAC operator's tool.

### 3.2.1 Security and User Profile Variables

The access to the QAMUT is limited to persons authorized to update metadata for certain data types only. QAMUT uses key information about authorized users. For each user login to JEST, JEST accesses the list of authorized QAMUT users and compares it with the logonid of its current users. The "QA UPDATE" button will be visible on the JEST's inspect results screen only to users who have permission to update Science or Operational QA metadata.

The minimum set of required attributes to implement MUT security and the format of the user profile file are as follows:

Logonid <tab> QAUupdate <tab> e-mail address <tab> DAAC e-mail address <tab> ShortName

The logonid is the user's ECS system logonid, QAUupdate has three valids {Science, Operational, both}. The user can be authorized to update Science QA metadata or Operational QA metadata or both. The Figure 3.2.1 illustrates interfaces between user authorization list and JEST.

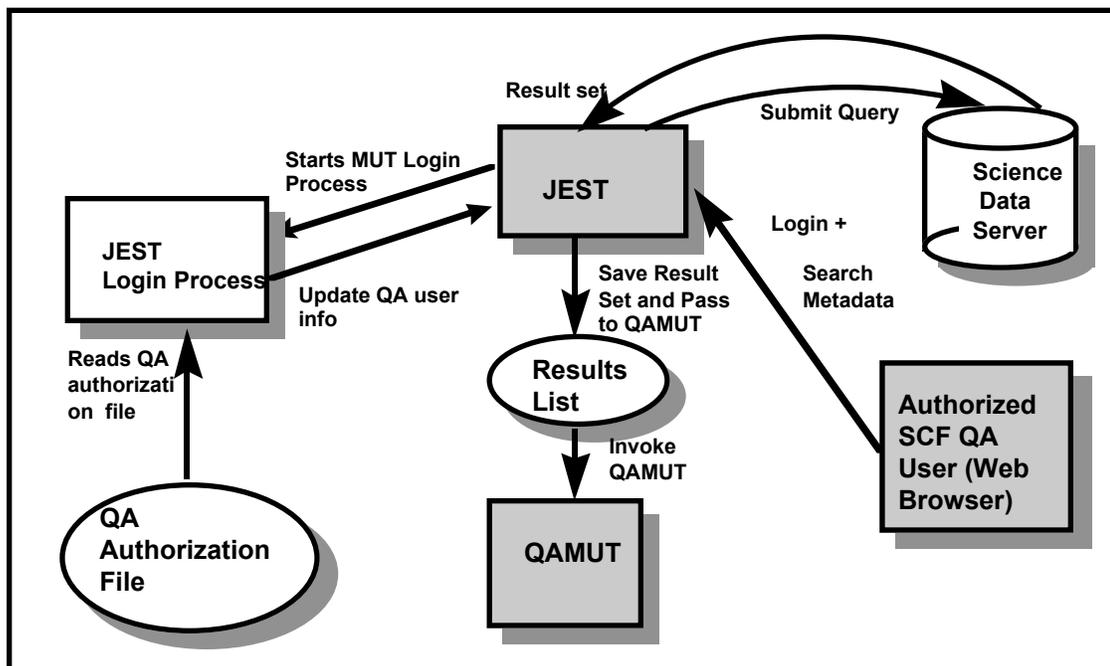
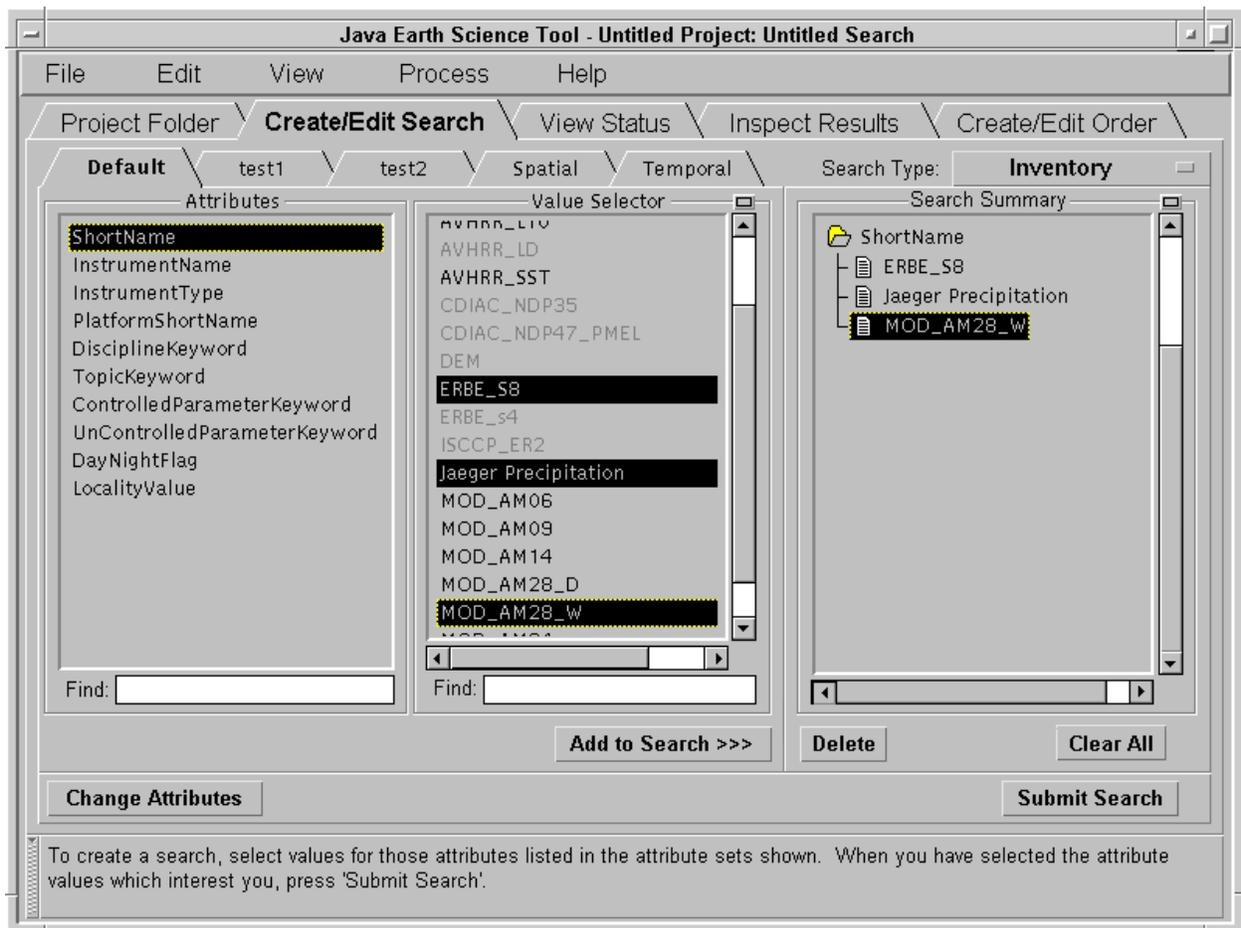


Figure 3.2.1. User Authorization for QA Update

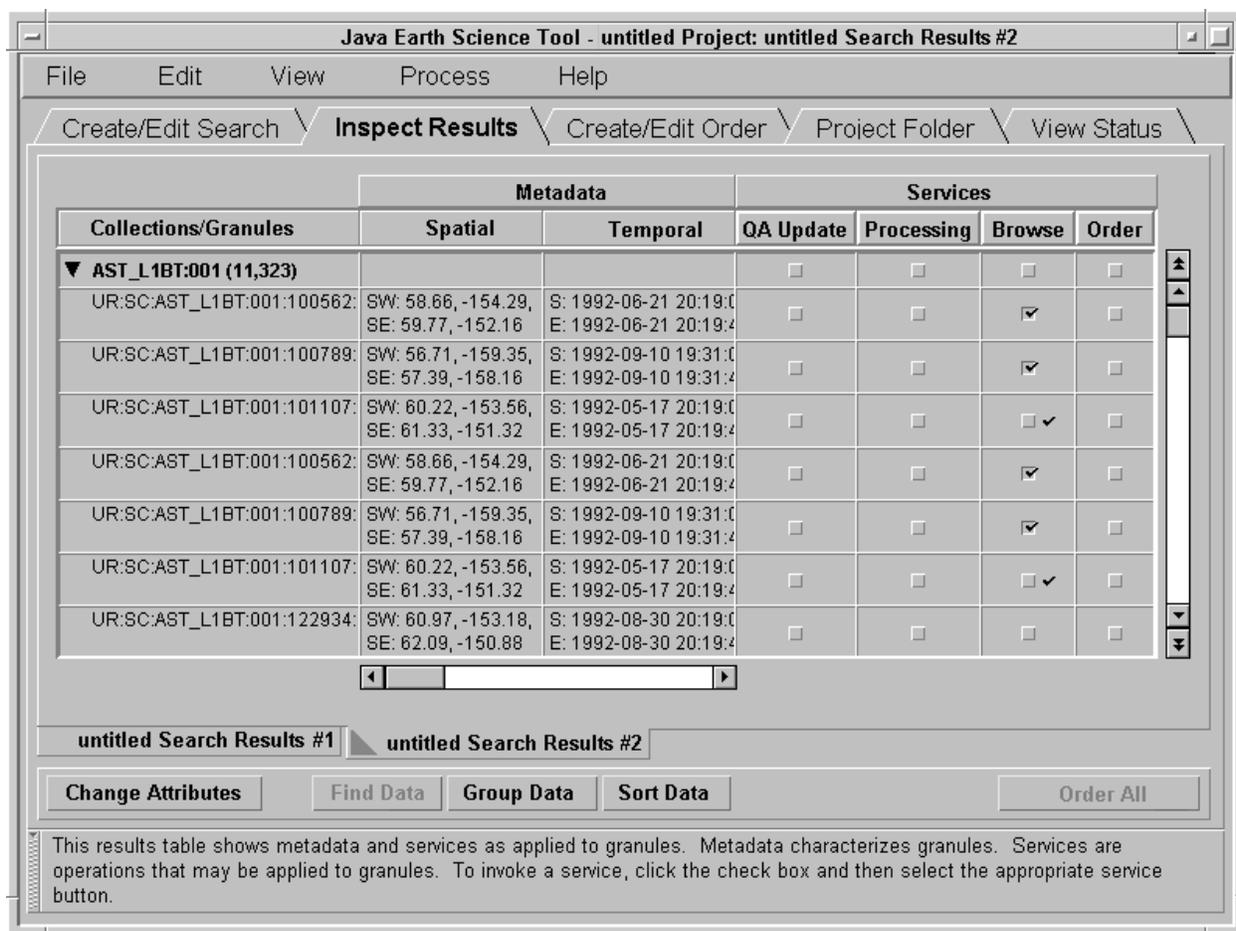
### 3.2.2 Search Query Through JEST

JEST will provide the search capability required for QA Metadata Update. The list of granules will be the results set resulting from the execution of the query based on the user selected search criteria entered in JEST. Users are responsible for setting their selection criteria appropriately, to yield the correct UR lists for QA metadata update.

JEST will be modified to interface with QAMUT and to provide access to QAMUT to authorized QA updaters only. The Graphical User Interface of the JEST will be modified to incorporate “QA UPDATE” button and QA check boxes. These button and boxes will be visible only to the users authorized to perform QA updates. The QA checkboxes will be displayed along with each granule in the data table so that users can selectively check granules for QA or select all granules resulting from the search query. The resulting set of granules will be saved in a file. The users will then click on the “QA UPDATE” button to invoke QAMUT and transmit the result to QAMUT for QA updates. The control will be returned to JEST inspect results screen when user exits QAMUT. The Figure 3.2.2A and Figure 3.2.2B show the JEST search query and resulting result screens layout.



**Figure 3.2.2A. The JEST Search Query Screen Layout**



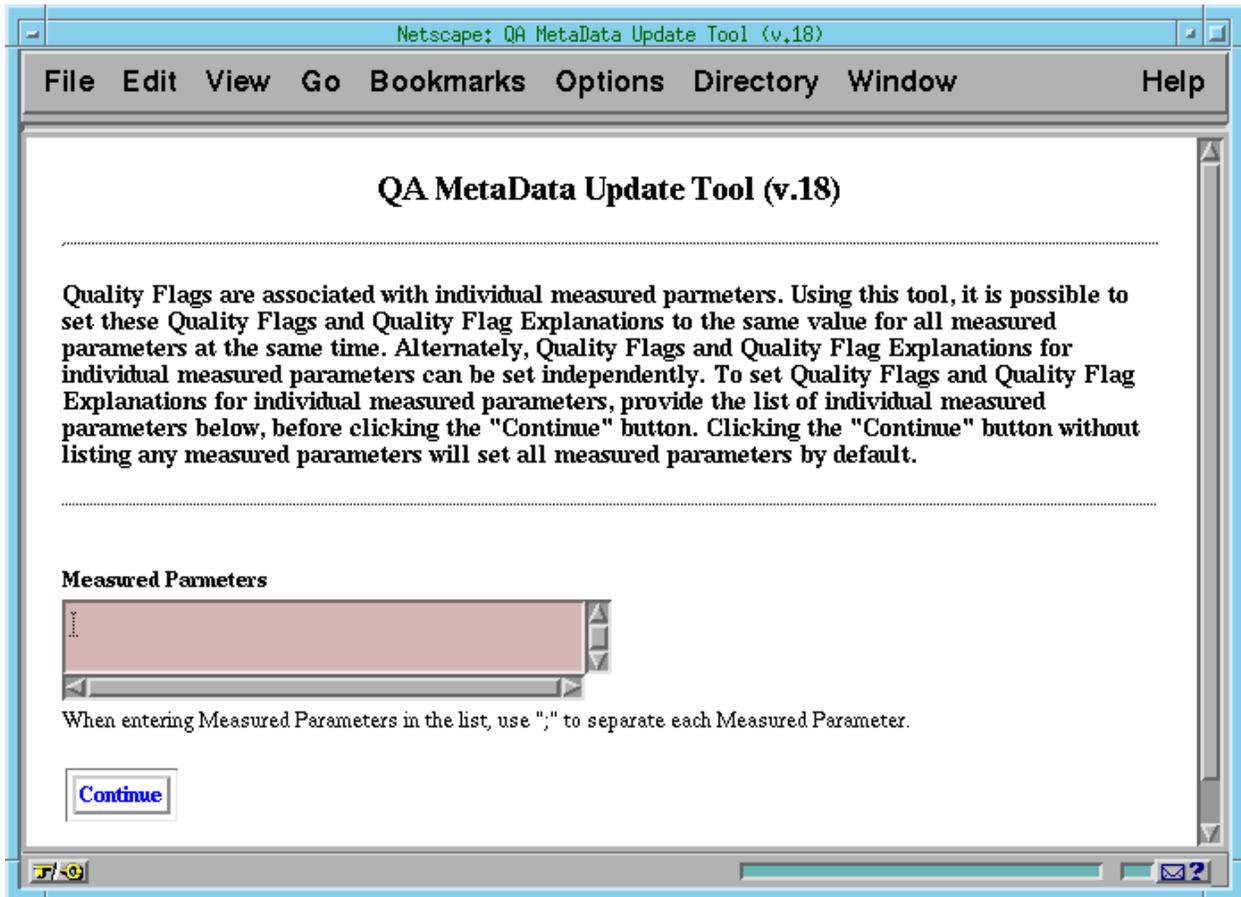
**Figure 3.2.2B. The JEST Query Results Screen Layout**

To aid users in identifying the different granules, each user-defined JEST results set should carry other metadata. Such as local granule ID, time/date, location, etc., or as specified by the user in JEST. If s/he will be updating the quality flags differently for multiple Measured Parameters within each granule, the user will need to know the MeasuredParameter for his/her data products, since JEST does not support the search on MeasuredParameter at launch.

### 3.2.3 QA Metadata Updates

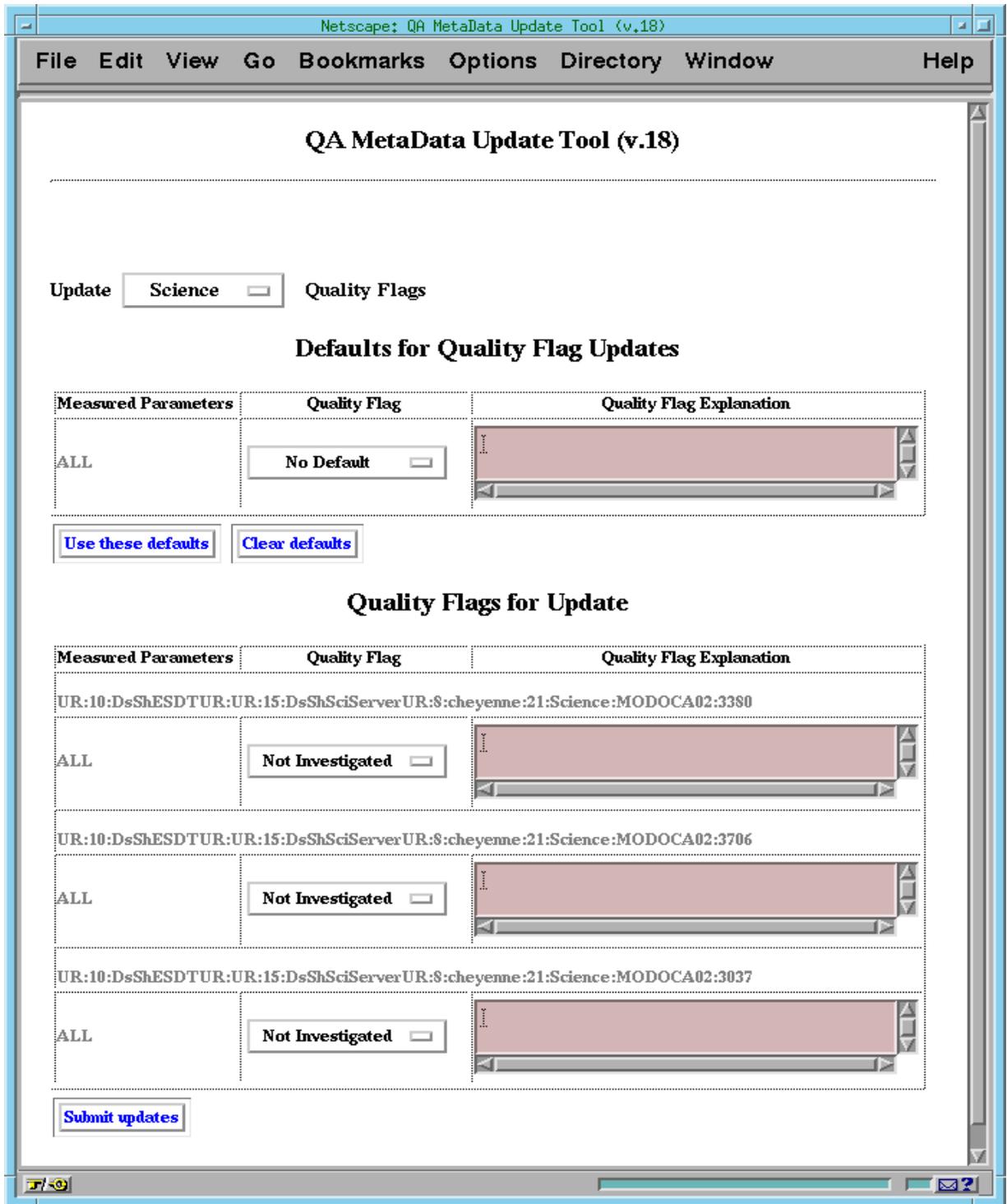
Based on B.0 model, each granule can contain multiple geophysical parameters. If the measurements were, say, two dimensional maps of the earth, then geophysical parameter is the third dimension. At the granule level, geophysical parameters are denoted as one or more "MeasuredParameter", each one of which has the full complement of QAFlags and, if appropriate, QAStats.

Once user clicks on the "QA METADATA" button, the SCF QA Metadata Update GUI will be invoked. The records resulted from the query from the previous screen will be displayed and available to user to perform QA. The information will be displayed sequentially on the screen for user to see. Based on user's privileges in the user profile, users can update either to ScienceQualityFlag, or OperationalQualityFlag. The Figure 3.2.3A shows the initial screen layout for the MUT interface.



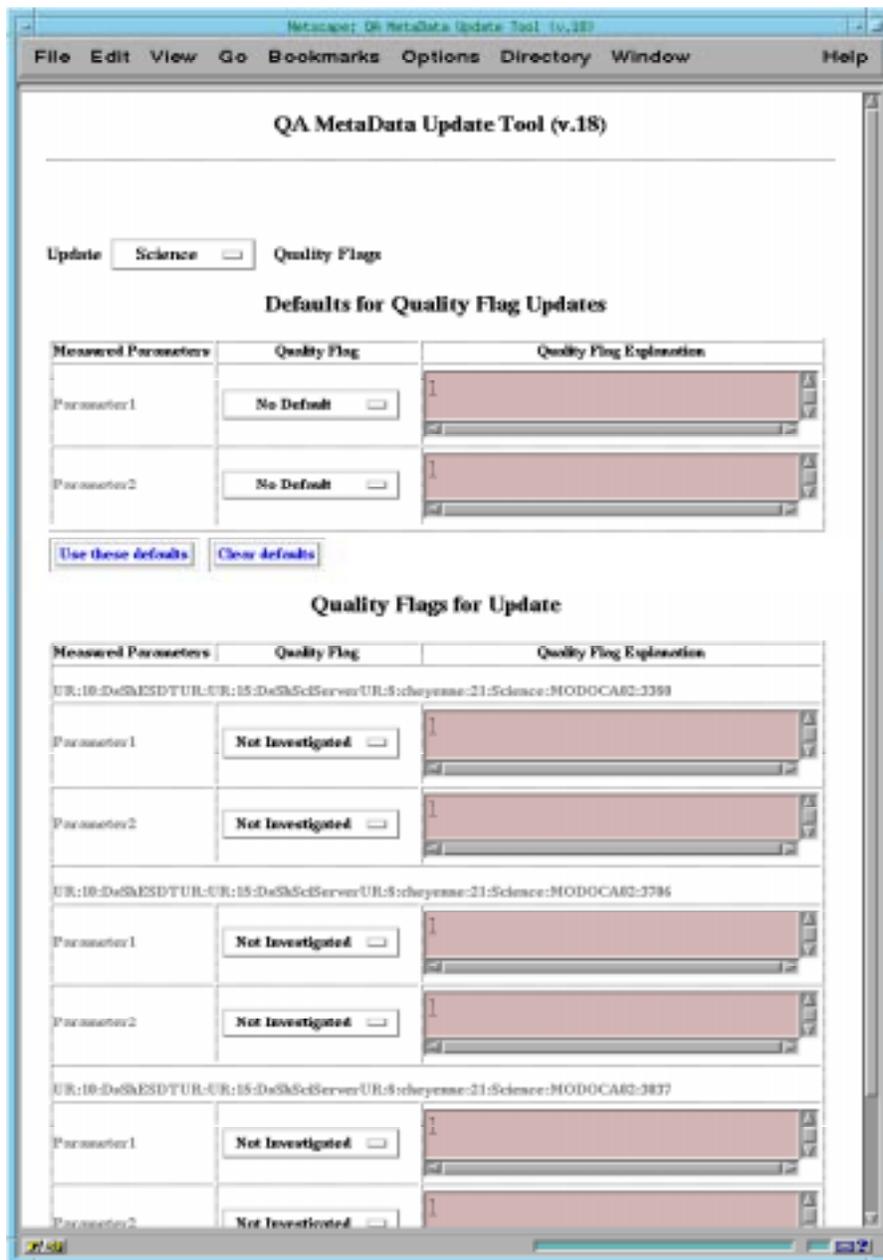
**Figure 3.2.3A. The QAMUT User Screen Layout**

On this screen the user enters a list of measured parameters, if desired. By not providing a list of measured parameters, the user by default will update all measured parameters for each granule. Clicking the continue button, brings up the screen to select the quality flag and quality flag explanation for the granules as shown in the Figure 3.2.3B.



**Figure 3.2.3B. The QAMUT User Screen Layout**

The QA metadata update on all displayed data will be performed by selecting a value from a pull-down menu. The tool would allow updates to all records in the result set by substituting user-specified valid value for the existing ScienceQualityFlag values. Presently, the domain of valid values for the ScienceQualityFlag include "Passed", "Failed", "Being Investigated", "Not Investigated", "Inferred Passed" and "Inferred Failed". A desirable feature of the tool is that it allows users to modify that single value to exceptions for those granules for which the standard value is incorrect. The Figure 3.2.3C shows the modified values.



**Figure 3.2.3C. The modified values to QA Metadata**

### 3.2.4 QA Updates Transmission

A “Forward QA Updates” button will be provided on the SCF QA Update interface. This button invokes the CGI e-mail program. The underlying code formats the message using the modified result set. The current format and content of the message is described below:

```
begin QAMetadataUpdate [nnnnn]
```

where nnnnn is the checksum to indicate the type of QA Metadata update, i.e., Science or operational update.

Following the begin line are lines of the form:

```
GranuleUR<tab>Measuredparametername<tab>Qaflagvalue<tab>Qaflagexplanationvalue
```

Fields in the above line are separated by tab characters. The QAFlagExplanationValue can be omitted if the user leaves it blank.

The MeasuredParameterName can be a valid Measured Parameter for the granule or it can be ALL. If more than one measured parameter is to be updated for a granule (and ALL is not used), separate lines should be included in the file for each parameter, repeating the granule UR.

For example:

```
From scientist@xxx.com Mon Oct 6 09:08 EDT 1997
Date: Mon, 6 Oct 1997 09:10:08 -0400
From: scientist@xxx.com
To: daacoperator@gsfc.nasa.gov
Subject: Metadataupdates
```

... some text ...

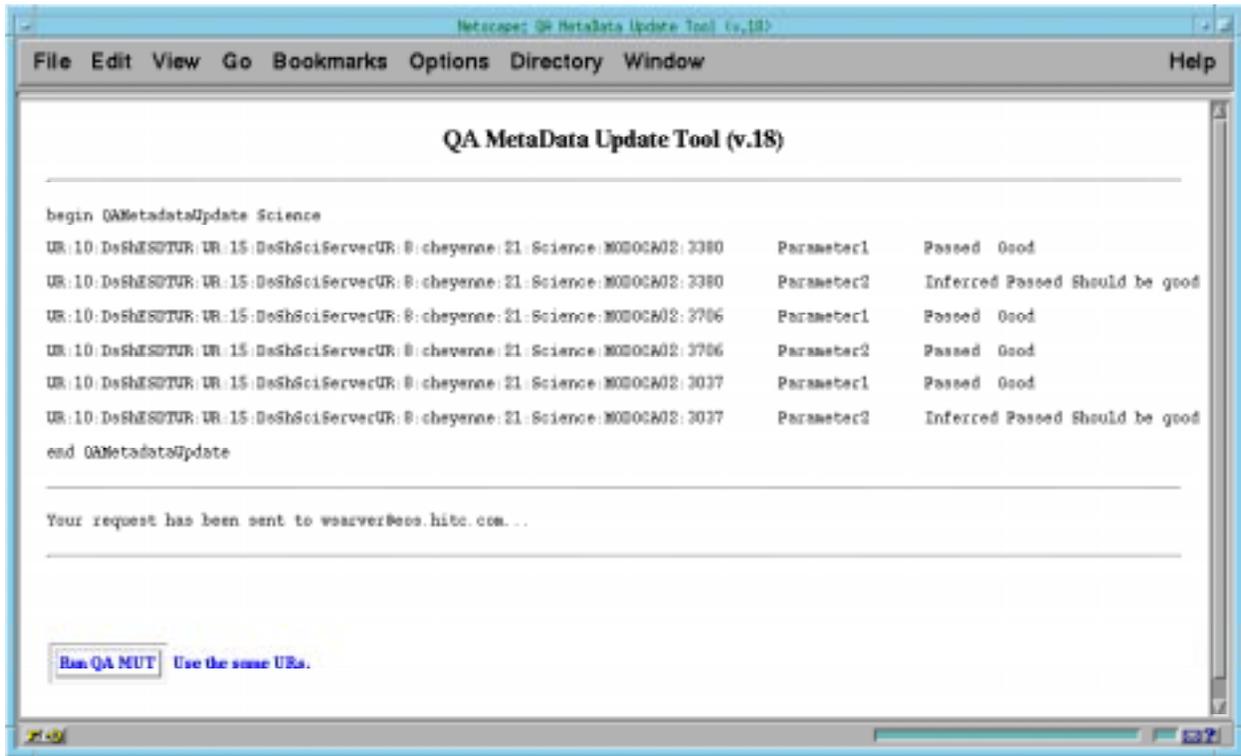
```
begin QAMetadataUpdate science
UR:10:DsShESDTUR:UR:15:DsShSciServerUR:8:cheyenne:21:Science:MODOCA02:3380<tab>param1<
tab>Passed<tab>some explanation
UR:10:DsShESDTUR:UR:15:DsShSciServerUR:8:cheyenne:21:Science:MODOCA02:3380<tab>param2<
tab>Failed<tab>some explanation
UR:10:DsShESDTUR:UR:15:DsShSciServerUR:8:cheyenne:21:Science:MODOCA02:3706<tab>ALL<tab>
>Passed<tab>
UR:10:DsShESDTUR:UR:15:DsShSciServerUR:8:cheyenne:21:Science:MODOCA02:3037<tab>ALL<tab>
>Passed<tab>
end QAMetadataUpdate
```

The update history is maintained for posterity and later may be used for trend analysis, if required. Any correction to unintended updates will require a new QA Update request submission to the DAAC Operator.

### 3.3 DAAC interface

The main objective of the DAAC operator tool is to actually perform the metadata updates as received from the SCF QAMetadata Tool. Specifically, the DAAC operator tool will parse the batch metadata update list provided by the SCF tool and initiate the changes in Science Data Server. The DAAC operator tool includes the functionality to go through Data Server interfaces to trigger any metadata update events.

The Figure 3-4 shows the update message as received by the DAAC operations. At the initiation of the tool, the operator views the list of metadata updates vs. granule and MeasuredParameter.



**Figure 3-4. The Update Message**

### 3.3.1 Security on the DAAC operator tool

Similar to the SCF user interface, the DAAC operator tool will adopt the same standards and the user will be required to have an authorized DAAC operator account. However, The DAAC operator tool is only available for use by the DAAC Operator.

### 3.3.2 User Notification

The DAAC operator tool waits until after the update then sends notification back to the user indicating the update was made. This not only provides the SCF with timely feedback, but also provides an independent communication mechanism between the SCF and the DAAC. In the event that someone other than the authorized user was inadvertently or maliciously using that user's account to update metadata, the authorized user could get timely feedback and take corrective action. The format of the message returned to the requester is included in Appendix

### **3.3.3 Update History**

The update history will be maintained by saving requested changes in an ASCII text file. Each batch update performed will be saved as a separate file. Each file name will include user name or loginid and datetime stamp. This would make the search, if needed, of these files easier at the DAAC side.

If requester wants to roll back or undo any changes then s/he will have to submit a new request.

This page intentionally left blank.

# Appendix A. QA Related Core Metadata from the B.0 Data Model

---

## Class = QAFlags

### AutomaticQualityFlag

The flag applying to measured parameters within a granule. When applied to parameter, the flag refers to the quality of that parameter for the granule (as applicable). The parameters determining whether the flag is set are defined by the developer and documented in the Quality Flag Explanation.

Content Source: PGE

Alias:

Constraints: One flag from QAFlags must exist.

Constraints: AutomaticQualityFlag must always be set in PGEs.

Domain:

Passed - The granule (or parameter) has passed a specified automatic test.

Failed - The granule (or parameter) has failed a specified automatic test.

Suspect - May be okay; could not clearly define.

### Class

QAFlags

### AutomaticQualityFlagExplanation

A text explanation of the criteria used to set automatic quality flag; including thresholds or other criteria.

Domain:

Free Text

### Class

QAFlags

## **OperationalQualityFlag**

The flag applying to measured parameters within a granule. When applied to parameter, the flag refers to the quality of that parameter for the granule (as applicable). The OperationalQualityFlag is set based upon criteria defined by DAAC operations organization and documented in the QualityFlagExplanation.

Content Source: DAAC

Alias:

Constraints: One flag from QAFlags must exist.

Domain:

Passed - The granule (or parameter) has passed a specified operational test.

Failed - The granule (or parameter) has failed a specified operational test.

Being Investigated - The granule (or parameter) is suspect and being investigated using a operational test.

Not Investigated - The granule (or parameter) has not been investigated by DAAC operational staff.

Inferred Passed

Inferred Failed

**Class**

QAFlags

## **OperationalQualityFlagExplanation**

A text explanation of the criteria used to set operational quality flag; including thresholds or other criteria.

Domain:

Free Text

**Class**

QAFlags

## **ScienceQualityFlag**

The flag applying to measured parameters within a granule. When applied to parameter, the flag refers to the quality of that parameter for the granule (as applicable). The ScienceQualityFlag is set based upon criteria defined by the developers and documented in the Quality Flag Explanation.

Content Source: DAAC

Alias:

Constraints: One flag from QAFlags must exist.

Domain:

Passed - The granule (or parameter) has passed a specified science test.

Failed - The granule (or parameter) has failed a specified science test.

Being Investigated - The granule (or parameter) is being investigated by an expert.

Validated - The granule (or parameter) has been validated by an expert.

Not Investigated - The granule (or parameter) has not been investigated by an expert.

Inferred Passed

Inferred Failed

**Class**

QAFlags

### **ScienceQualityFlagExplanation**

A text explanation of the criteria used to set science quality flag; including thresholds or other criteria.

Content Source: Data Producer

Alias:

Domain:

Free Text

**Class**

QAFlags

This page intentionally left blank.

# Appendix B. Format of the Message Returned to the Requester

---

From daacoperator@gsfc.nasa.gov Mon Oct 6 09:08 EDT 1997  
Date: Mon, 6 Oct 1997 09:10:08 -0400  
To: scientist@xxx.com  
Subject: Results of Metadata Update

X-UIDL: f32416560ed174b450708339e7c2e251

Input requests:

UR:10:DsShESDTUR:UR:15:DsShSciServerUR:8:cheyenne:21:Science:MODOCA02:3380 param1

Passed an explanation

UR:10:DsShESDTUR:UR:15:DsShSciServerUR:8:cheyenne:21:Science:MODOCA02:3706 param1

Failed

UR:10:DsShESDTUR:UR:15:DsShSciServerUR:8:cheyenne:21:Science:MODOCA02:3706 param2

Passed

UR:10:DsShESDTUR:UR:15:DsShSciServerUR:8:cheyenne:21:Science:MODOCA02:3037 ALL

Passed some explanation

End of input requests

Processing updates for

UR:10:DsShESDTUR:UR:15:DsShSciServerUR:8:cheyenne:21:Science:MODOCA02:3380

Granule metadata before update:

-UnnamedPL[

--MeasuredParameter[

---MeasuredParameterContainer[ParameterName(ParameterName)

----QASStats[QAPercentMissingData(0) QAPercentOutOfBoundsData(0)

QAPercentInterpolatedData(100) QAPercentCloudCover(50)]

----QAFlags[AutomaticQualityFlag(Passed) AutomaticQualityFlagExplanation(1)

OperationalQualityFlagExplanation(OperationalQualityFlagExplanation)

ScienceQualityFlag(Passed)

ScienceQualityFlagExplanation(ScienceQualityFlagExplanation)]]]]

Updating QAFlags for measured parameter param1

MeasuredParameter param1 not found in granule metadata

Errors found during update process. Updates not done for this granule.

Processing updates for

UR:10:DsShESDTUR:UR:15:DsShSciServerUR:8:cheyenne:21:Science:MODOCA02:3706

Granule metadata before update:

-UnnamedPL[

--MeasuredParameter[

---MeasuredParameterContainer[ParameterName(ParameterName)

----QASStats[QAPercentMissingData(0) QAPercentOutOfBoundsData(0)

QAPercentInterpolatedData(100) QAPercentCloudCover(50)]

----QAFlags[AutomaticQualityFlag(Passed) AutomaticQualityFlagExplanation(1)

OperationalQualityFlagExplanation(OperationalQualityFlagExplanation)

ScienceQualityFlag(Failed)

ScienceQualityFlagExplanation(ScienceQualityFlagExplanation)]]

---MeasuredParameterContainer[ParameterName(param1)

----QASStats[]

----QAFlags[]]

---MeasuredParameterContainer[ParameterName(param2)

----QASStats[]

----QAFlags[]]

---MeasuredParameterContainer[ParameterName(param3)

----QASStats[]

----QAFlags[ScienceQualityFlag(zonk)]]]]

Updating QAFlags for measured parameter param1

Updating QAFlags for measured parameter param2

```
-----
Detailed result status of update from data server:
-ReqResults[
--CmdResults[
---ESDTResults[(Updated metadata failed validation.) ErrorMessage(ESDT Execution failed)]
ESDTStatus(0) CmdSuccess(1)]
-----
```

```
Granule metadata after update:
-UnnamedPL[
--MeasuredParameter[
---MeasuredParameterContainer[ParameterName(ParameterName)
----QASStats[QAPercentMissingData(0) QAPercentOutOfBoundsData(0)
QAPercentInterpolatedData(100) QAPercentCloudCover(50)]
----QAFlags[AutomaticQualityFlag(Passed) AutomaticQualityFlagExplanation(1)
OperationalQualityFlagExplanation(OperationalQualityFlagExplanation)
ScienceQualityFlag(Failed)
ScienceQualityFlagExplanation(ScienceQualityFlagExplanation)]
---MeasuredParameterContainer[ParameterName(param1)
----QASStats[]
----QAFlags[ScienceQualityFlag(failed) ScienceQualityFlagExplanation(failed)]
---MeasuredParameterContainer[ParameterName(param2)
----QASStats[]
----QAFlags[ScienceQualityFlag(passed) ScienceQualityFlagExplanation(passed)]
---MeasuredParameterContainer[ParameterName(param3)
----QASStats[]
----QAFlags[ScienceQualityFlag(zonk)]]]]
```

```
Processing updates for
UR:10:DsShESDTUR:UR:15:DsShSciServerUR:8:cheyenne:21:Science:MODOCA02:3037
```

```
Granule metadata before update:
-UnnamedPL[
--MeasuredParameter[
---MeasuredParameterContainer[ParameterName(ParameterName)
----QASStats[QAPercentMissingData(0) QAPercentOutOfBoundsData(0)
QAPercentInterpolatedData(100) QAPercentCloudCover(50)]
----QAFlags[AutomaticQualityFlag(Passed) AutomaticQualityFlagExplanation(1)
OperationalQualityFlagExplanation(OperationalQualityFlagExplanation)
ScienceQualityFlag(passed) ScienceQualityFlagExplanation(some explanation)]]]]
Updating QAFlags for measured parameter ALL
```

```
-----
Detailed result status of update from data server:
-ReqResults[
--CmdResults[
---
ESDTResults[UR(UR:10:DsShESDTUR:UR:15:DsShSciServerUR:8:cheyenne:21:Science:MODOCA02:3
037)] ESDTStatus(1) CmdSuccess(1)]
-----
```

```
Granule metadata after update:
-UnnamedPL[
--MeasuredParameter[
---MeasuredParameterContainer[ParameterName(ParameterName)
----QASStats[QAPercentMissingData(0) QAPercentOutOfBoundsData(0)
QAPercentInterpolatedData(100) QAPercentCloudCover(50)]
----QAFlags[AutomaticQualityFlag(Passed) AutomaticQualityFlagExplanation(1)
OperationalQualityFlagExplanation(OperationalQualityFlagExplanation)
ScienceQualityFlag(passed) ScienceQualityFlagExplanation(some explanation)]]]]
```

# Abbreviations and Acronyms

---

DAAC	Distributed Active Archive Center
ECS	EOSDIS Core System
IT	Instrument Team
MUT	Metadata Update Tool
QA	Quality Assessment
SCF	Science Computing Facility