NATCDC & NATCDCSP

PLOG Processing Solutions for ADABAS

Installation & Operation Manual

For

All Platforms

NatWorks Incorporated
454 South Main Street
Northfield, VT 05663
(802) 485-6112

Website: www.natworks-inc.com
E-Mail: info@natworks-inc.com

Table of Contents

Introduction

NatWorks, Inc. is very pleased to have you as a user of NATCDC / NATCDCSP, and we are confident that you will find NATCDC / NATCDCSP both powerful and easy to use.

This document applies to NATCDC Version 1.9.1 and NATCDCSP 4.2.0 all subsequent versions, unless indicated in new editions, technical newsletters or release notes. Specifications contained herein are subject to change, and these changes will be appropriately reported in subsequent revisions or editions.

ADABAS PLOG Processing Overview

Within a production Software AG ADABAS environment, it is the usual mode of operation to run ADABAS with a Protection Log (PLOG). The primary purpose of the PLOG is to provide a mechanism through which ADABAS transactional data can be recovered in the event of a system failure.

Simplistically, the PLOG works by recording all transactional changes that occur into a sequential dataset. With this transactional data captured, various ADABAS utilities allow this data to then be utilized with and against ADABAS to address various needs of ADABAS recovery.

While the primary use of PLOG data is to serve as a data source for data recovery, the data that a PLOG contains has other potentially beneficial uses to an organization, such as:

- Capturing ADABAS transactional data for updating Data Warehouses
- Audit Reporting
- Application Troubleshooting

In examining how PLOG data is stored, it becomes evident that there are obstacles that present themselves in attempting to process this PLOG data so that it can be easily used in other ways. Briefly, these obstacles would include:

1. **A PLOG contains transactions against all ADABAS files within the source database**
   In order to examine / process transactions occurring against individual files, it is generally required to "split" these transactions into separate files that only contain transactions for a given file of interest.

2. **PLOG data is stored in compressed form**
   PLOG data reflects the manner in which data is stored into ADABAS in that the transactional records are stored in compressed format. In order to process any data contained in a PLOG, it is eventually required that the transactional records must be decompressed.

3. **PLOG data is likely to be stored as variable-length records**
   Due to the fact that ADABAS handles "unique" data structures such as Periodic-Groups (PEs) and Multi-Valued fields (MUs), and given that the number of occurrences for these types of fields can change between records; the lengths of transactional records for a given file of interest can vary from record to record within the same transactional file.

4. **PLOG data may need translation**
   While ADABAS can operate on several different platforms; the historical platform of ADABAS is a mainframe environment that is EBCDIC-based. In most cases, data from ADABAS is required to be placed upon ASCII-based platforms, and this requires that EBCDIC data types be transferred into ASCII-compatible equivalents. Even when ADABAS resides on an ASCII-based platform, ADABAS contains some unique data formats that must be translated into more usable formats (such as Date, Time and Timestamp fields).

To address the first and second issues noted above, NatWorks solutions utilize the ADABAS utility ADASEL or ADACDC in mainframe platforms or ADAPLP in UNIX, Linux and Windows platforms. These utilities primarily handle the decompression of transactional records, and these utilities also have varying abilities of "splitting off" transactions for files of interest from other PLOG data that is not relevant or required.

To address the third issue noted above, NatWorks designed and developed NATCDC: A PLOG-processing solution that resolved these difficulties in a manner that is straight forward, high-performance, and completely native to a Software AG environment. NATCDC however was only designed to handle a single file at a time, meaning that if the need was to provide PLOG processing against 100 ADABAS source files, there would by necessity have to be 100 separate NATCDC processes to process each of the desired files. While all of these processes could execute concurrently, or alternatively all 100 processes could be job-streamed together; this could not help but be seen as a limitation with NATCDC. To address that limitation, NatWorks enhanced NATCDC processing such that it could handle any number of files in a single processing pass, meaning that a single process could handle 100 (or more) source file processes in a single pass execution. NatWorks named this enhanced processing, NATCDCSP.

Both NATCDC and NATCDCSP essentially provide the same functioning. For all records in a given file, NATCDC / NATCDCSP will convert what is likely to be variable-length records into fixed-length format. NATCDC and NATCDCSP also have the optional ability to drop any un-needed fields from records, leaving only those fields of interest.

To address the fourth issue, NatWorks has extended the generational capabilities of NatQuery to generate custom Natural programs that operate to deliver transactional data into a form that supports DWH processing in several ways.

Benefits of Using NATCDC / NATCDCSP

The combination of NatQuery, existing ADABAS utilities, and NATCDC / NATCDCSP offers existing Software AG customers an extremely powerful, straightforward, flexible, easily understood, and cost-effective solution to the problem of tapping the data available in a PLOG.

Using NATCDC / NATCDCSP provides a customer with the following benefits:

- **A Completely Native Solution**
  By using NATCDC / NATCDCSP, nothing "foreign" is introduced into a SAG environment. Existing ADABAS utilities are used for what they were designed to do, and Natural processing is used to provide all remaining functionality.

- **Performance of Decompression**
  NATCDC / NATCDCSP provides one of the best performing solutions, both in terms of CPU usage and Clock Time, due to the simple fact that the most resource-intensive process involved with handling PLOG data is decompression. Since Software AG wrote the compression algorithms for ADABAS, it makes sense that Software AG's might best understand the processing needed to decompress – and these utilities are both fast and completely trustworthy.

- **Flexibility**
  NATCDC / NATCDCSP supports DWH processing in a number of ways, including the ability to:

  - Generate "interface" files that describe the layout of converted PLOG output so that ETL tools can immediately consume the converted PLOG output.

  - Provide direct integration of PLOG output into popular RDBMS targets such as SQL Server and Oracle, with direct integration to other targets being developed.

  - Provide PLOG output that is immediately usable in a manual fashion for manually-handled processing into Relational Database Management Systems (RDBMS).

  Beyond the abilities to support DWH initiatives, NATCDC / NATCDCSP can also be used to support transaction auditing and transaction analysis.

- **Version Protection**
  Whenever Software AG should decide to make internal changes to the current structure of ADABAS PLOGS, an approach that is based on ADABAS utility output is far better insulated from changes across versions than Any Other Method. To maintain backwards compatibility, Software AG always works to insure that utilities operate consistently even across versions – and NATCDC / NATCDCSP take full advantage of this.

- **Return on Investment**
  As opposed to alternative solutions that provide decompression within their own software, essentially forcing a customer to pay for technology that is redundant to technology already present within ADABAS utilities, NATCDC / NATCDCSP utilize the processing that a Software AG customer has already bought and paid for.

  Further, with all required processing not handled with ADABAS utilities being handled with 100% generated Natural – the delivered final processing package will be immediately understood as to how it operates. It's only Natural…

- **Performance in Overall Execution**
  Both NATCDC and NATCDCSP were created with an eye on achieving the best performance possible when performing the tasks required.

  For mainframe environments, additional performance gains are achieved through the fact that NATCDC and NATCDCSP are delivered in Object Code form, with this Object Code having been compiled using the Software AG Natural Optimizer Compiler (NOC) product. This results in Object Code that is optimized for mainframes to the highest possible extent, reducing execution time by as much as 33% over the execution of a non-NOC compiled version of the same program. While NATCDC and NATCDCSP are compiled under the Natural Optimizer Compiler: There is no requirement that a user of NATCDC or NATCDCSP must have a license for the Natural Optimizer Compiler for these programs to operate as intended.

  Software AG currently does not provide a Natural Optimizer Compiler designed for the UNIX, Linux or Windows environments.

General Features

NATCDC and NATCDCSP both offer the following general features:

- **100% data integrity**
  - PLOGs represent the source of ADABAS restart and recovery and both NATCDC and NATCDCSP accurately and efficiently interpret and convert PLOG transactions into usable information

- **Simple Mainframe Installation**
  - 1 single Natural Processing Program (NATCDC or NATDCDSP)
  - 1 Generated Natural Processing Program per Source File (NATCDC)
  - 1 Generated Traffic Program and ***n*** Natural Processing Subprograms – one for each Source File (NATCDCSP)

- **Full handling of all ADABAS data structures including**:
  - Multi-Valued Fields (MUs), Periodic-Group Fields (PEs) and MUs in PEs
  - Automatic format translations including Date, Time and Timestamp
  - Sign handling of all numeric-based fields
  - Capture of C* values for all MUs, PEs and MUs in PEs
  - Ability to apply user-specified Edit Masks for the output of specific fields

- **Efficient and Trusted Decompression**
  - ADASEL, ADACDC and ADAPLP are Software AG-supplied utilities built specifically to handle the decompression of PLOGs

- **Conversion of variable-length records to fixed-length**
  - If a given source file contains any recurring field such as MUs, PEs or MUs in PEs; then these will generally be reflected as variable-length records in a raw PLOG or the output of ADABAS PLOG utilities. NatCDC and NATCDCSP convert these variable-length records to fixed length, as fixed-length data is easier and more efficient to handle.

- **PLOG record header translation**
  - PLOG records contain a standard header that contains transactional information pertaining to when and what caused the transaction, and the NATCDC / NATCDCSP process converts key fields in this header into usable formats.

- **Full Sensitivity to ISN Reuse**
  - NATCDC / NATCDCSP processing is sensitive to the fact that a record with an ISN can be Deleted, with a new record being subsequently Stored that is given the same ISN value due to ISN re-usage.

- **Full and complete handling of all transactions, including back-outs**
  - The NATCDC / NATCDCSP process directly reflects the processing of ADABAS itself. As a result of this, NATCDC / NATCDCSP deliver a true "Point in Time" snapshot of ADABAS. Any and all updates into ADABAS, including ADABAS back-outs or application back-outs are fully and completely handled.

- **Ability to select / drop specific fields in final output**
  - The NATCDC / NATCDCSP administrator can easily select which fields are of interest to PLOG processing, and can drop those fields that are not of interest from the final output.

- **Support for expanded files, with physical to logical ISN conversion**
  - NATCDC / NATCDCSP fully support ADABAS "expanded" files, and converts a physical ISN into a logical one.

- **Full generation of all objects required for PLOG handling**
  - Fully Graphical User Interface (GUI) of NatQuery / NATCDC allows the entire NATCDC / NATCDCSP process to be generated.

- **Automatic FTP of generated objects to Mainframe Server**
  - If desired, NATCDC / NATCDCSP can automatically FTP all objects (parameters, programs and JCL) directly onto the server platform (mainframe, UNIX, Linux or Windows).

- **Selectable Options to support for final output:**
  - All Transactions
  - Logical "Last" image (flagged as **S**tore, **D**elete or **U**pdate)
  - Logical "First and Last" images (flagged as first **B**efore and Last **A**fter)

- **Reporting**:
  - Exception reporting for recurring field handling
  - Total count of all Before and After images
  - Total of all records handled
  - Total number of Stores, Deletes, Updates and Store / Deletes
  - Total number of unique records handled

- **Ability to process PLOG transactions against a Single File or Multiple Files**
  - The use of NATCDC allows for the execution of a single process that will properly handle the PLOG transactions for a single file.

  - The use of NATCDCSP allows for the execution of a single process that will properly handle the PLOG transactions from any number of files in a single pass.

NATCDC / NATCDCSP Pre-Requisites

The following should be considered as pre-requisites for NATCDC / NATCDCSP processing.

1. **NatQuery must be Installed**
   While NatCDC / NATCDCSP can be purchased separately from NatQuery, then the purchasing organization will still receive NatQuery although the full use of NatQuery will be limited though the License Key provided.

   In order for NATCDC / NATCDCSP to function as designed, NatQuery must be installed to provide the generation and administration functions that NATCDC / NATCDCSP requires. In order to perform these functions, NatQuery is provided in a limited version as part-and-parcel of the stand-alone purchase of NatCDC or NatCDCSP.

2. **NatQuery should be configured for FTP / File Copy (optional)**
   If automatic uploading of generated processes is desired, NatQuery should be first properly configured to use FTP / File Copy.

3. **NatQuery should be configured for required DDMs**
   In order to operate as intended, NATCDC / NATCDCSP requires that specific information be captured into NatQuery: Information such as DDMs for required files and information relating to those DDMs such as **Occurrence Information**, **Descriptor Statistic Information**, possibly **Sign Byte Information** and specific **JCL / Scripts** templates.

   While this document does briefly describe the configuration process required in NatQuery in order to facilitate NATCDC / NATCDCSP generation, this information is not as in-depth as what is provided in the *NatQuery Installation and Operations* manual.

4. **Protection Log Availability**
   In order to function as designed, the site must be running ADABAS with Protection Logs (PLOGs) turned on, and the designated NATCDC / NATCDCSP Administrator must be given access to these files.

5. **Natural NATLOAD utility must be available** (Mainframe Environments)
   In order to move the core NATCDC / NATCDCSP processing module into a mainframe Natural server environment, the Natural utility NATLOAD is used. This means that this utility must be available, and if Natural Security is installed, the designated NATCDC / NATCDCSP Administrator must have access to this utility.

6. **Natural SYSOBJH utility must be available** (UNIX, Linux and Windows)
   In order to move the core NATCDC / NATCDCSP and NATPLP processing modules into a UNIX, Linux or Windows Natural server environment, the Natural utility SYSOBJH is used. This means that this utility must be available to the installer, and if Natural Security is installed the designated NATCDC / NATCDCSP Administrator must have access to this utility.

7. **Availability of a Server User-ID and Associated Password**
   In order to properly install a NATCDC / NATCDCSP process, the designated NATCDC / NATCDCSP administrator should have access to the Natural / ADABAS server platform and should be granted FTP / network access to this platform. If Natural Security is installed, then a Natural Security User-id and password must be available that will allow execution of the NATLOAD or SYSOBJH utility.

8. **Creation of a Natural Library Named NATCDC / NATCDCSP**
   If Natural Security is installed a library called NATCDC or NATCDCSP should be defined. This library must be defined to minimally allow full access to the designated Administrator's Natural User-ID.

9. **Familiarity with Mainframe Dataset Allocation Utilities** (Mainframe Environments)
   In order to install the NATCDC / NATCDCSP processing programs, they must first be moved from the workstation environment into a dataset(s) defined with precise attributes.

   It is therefore a requirement that the individual who is installing the NATCDC software be able to use mainframe-based utilities to properly define this dataset.

10. **Familiarity with the Server's SORT utility**
    Part of the NATCDC / NATCDCSP processing requires the use of a server-based SORT program that can properly sort / handle binary data. The NATCDC / NATCDCSP Administrator should have some familiarity with this SORT utility in order to properly set up the required NATCDC / NATCDCSP JCL / Script template.

11. **License Key**
    In order to see or use the NATCDC / NATCDCSP functions of NatQuery, NatQuery must first be given a License Key for NATCDC, just as a License Key is required for normal NatQuery operation.

    This License Key can be entered into NatQuery when NatQuery is first executed, or input into NatQuery at a later time through the **NatQuery Configuration** function. To access the **NatQuery Configuration** function on an open NatQuery desktop, begin by clicking on the **Administer** drop-down menu and then click on **NatQuery Configuration**. This will invoke the **NatQuery Configuration** window. On the **NatQuery Configuration** window there are a series of Tabs, one of which will be labeled **License Key(s)**. To enter

a License Key for NATCDC, click on the **License Key(s)** tab, and then enter the NatWorks-provided NATCDC License Key into the appropriate fields. Click the **OK** button when done.

NATCDC / NATCDCSP Processing Overview

To assist the reader in understanding how NATCDC / NATCDCSP operate, the following sections graphically depict the processing steps for both NATCDC and NATCDCSP.

The determination of whether a NATCDC process is needed or whether a NATCDCSP process is needed is simple. If the need is to process a PLOG file for transactions that occurred against ***multiple*** ADABAS source files, then a NATCDCSP process is needed. If the need is to only process a PLOG file for transactions that occurred against a ***single*** ADABAS source file, then either a NATCDC process can be used (a process which is designed to handle only a single file's transactions), or a NATCDCSP process can be used (a process capable of processing multiple files but which is configured to only process a single file).

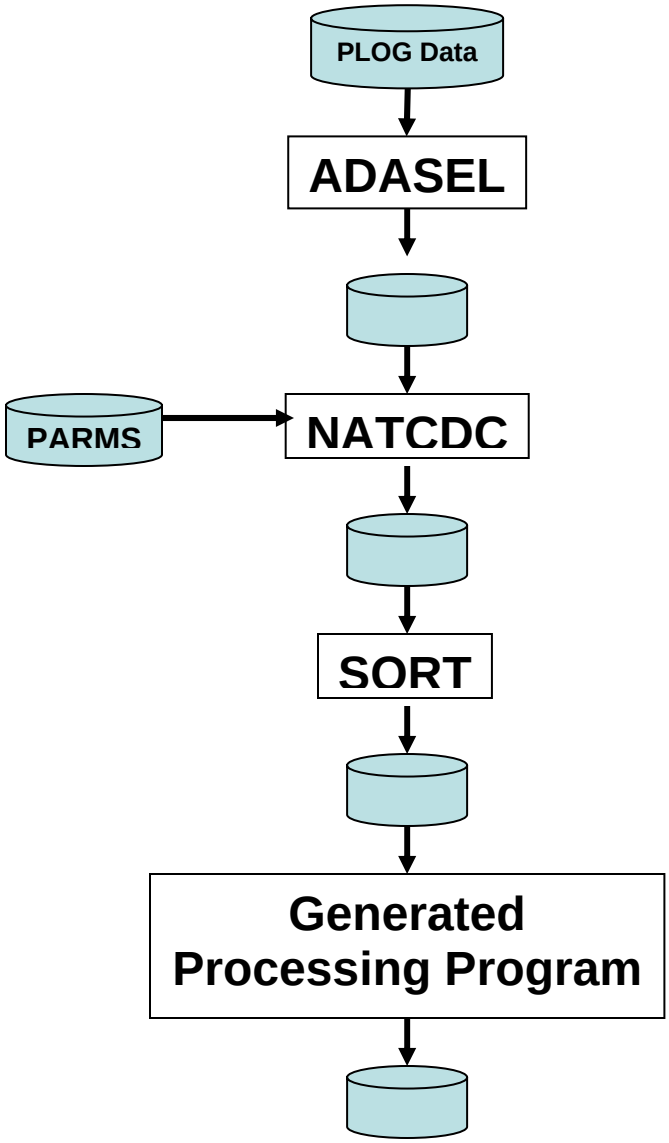
Since the processing is slightly different between Mainframe environments and UNIX, Linux or Windows ("Open Systems") environments, each type of processing for each platform is detailed in separate sections.

Available sections are:

- **Mainframe Processing**
  - NATCDC and Mainframe Processing
  - NATCDCSP and Mainframe Processing
- **UNIX / Linux / Windows Processing ("Open Systems")**
  - NATCDC and Open Systems Processing
  - NATCDCSP and Open Systems Processing

NATCDC and Mainframe Processing

The following graphic depicts the processing involved with the execution of NATCDC in a Mainframe Environment. A description of this processing follows below.



Process Description for NATCDC on Mainframe Platforms

The above graphic depicts the following processing:

1. **PLOG Data**
   Processing begins with raw PLOG data. PLOG data is written by ADABAS as variable length records, with a PLOG file containing all transactions that occurred against a specific ADABAS for the time period for which the PLOG was active.

2. **ADASEL**
   The ADABAS utility ADASEL is used to decompress the PLOG records of interest. The execution of the ADASEL utility provides the data from PLOG transactions in a variable-length sequential file, with the ADASEL output either containing transaction records from one single file or all files, based on the parameters used with ADASEL.

   As an alternative to the use of ADASEL, the ADABAS utility ADACDC can also be used – please contact NatWorks for further details on the use of ADACDC over ADASEL.

3. **NATCDC**
   The NatWorks utility NATCDC is used to process the output of ADASEL. NATCDC reads the variable-length sequential input file and selects the records which pertain to the given file of interest (NATCDC is designed to process only a single file in a single pass). When a record is selected, NATCDC converts the variable-length record into a fixed-length record, and it can optionally "drop" any fields which are not of interest to the user.

   NATCDC operates by receiving a NatQuery-generated parameter file, with this parameter file controlling how the fixed-length record should be created, and also dictating what (if any) fields should be dropped.

4. **SORT**
   The next step in the processing is the execution of a system SORT which will sort the records into ascending sequence by ISN and Transaction Sequence.

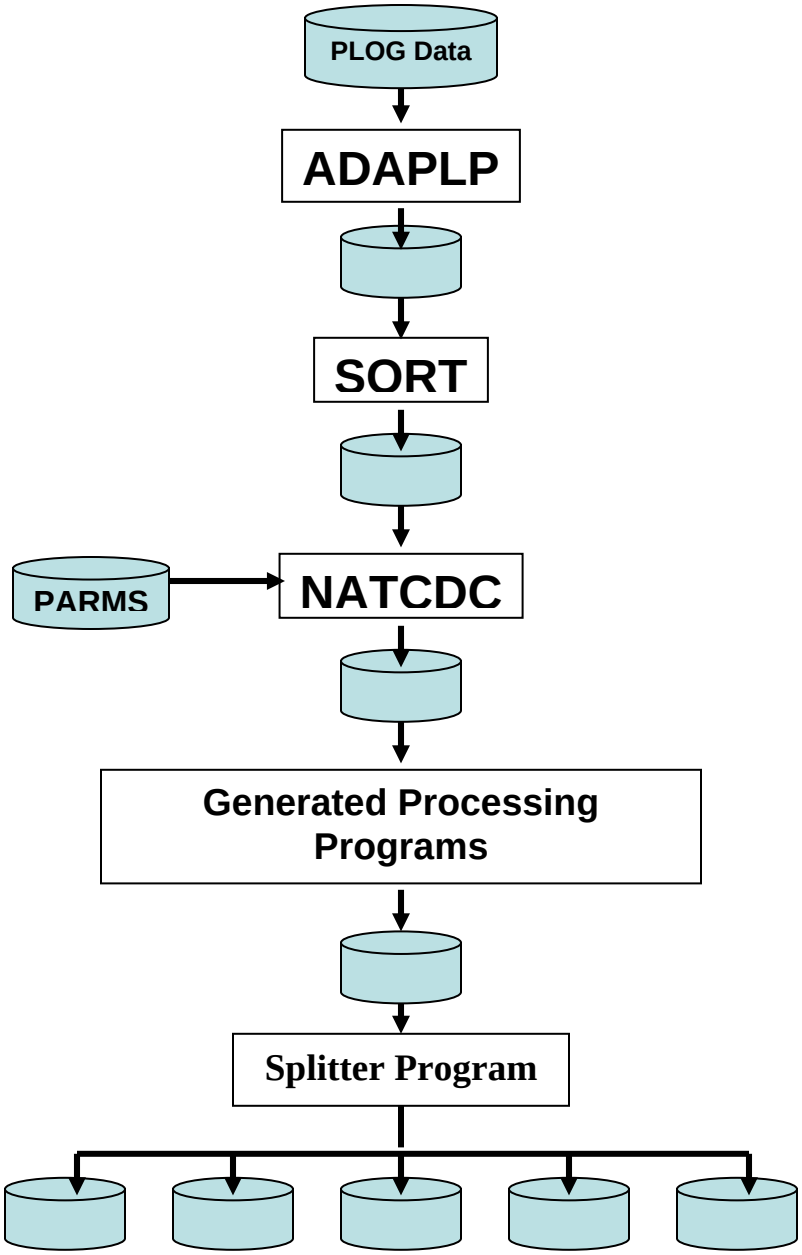
5. **Post-Processing Program**
   The final step in the processing is the execution of a generated Natural program that is specific to the file being handled. Optionally, this generated program can deliver "Delta" images, it can deliver the First and Last images for each ISN found, or it can deliver all images (pass through all images – suitable for auditing).

   The Post-Processing program will convert any and all "difficult" ADABAS data-types into ASCII equivalents, including properly handling Date, Time and Timestamp fields.

The post-processing program also produces numerous counts and totals in a report which details exactly what the post-processing handled and what it output.

NATCDCSP and Mainframe Processing

The following graphic depicts the processing involved with the execution of NATCDCSP in a Mainframe Environment. A description of this processing follows below.



Process Description for NATCDCSP on Mainframe Platforms

The above graphic depicts the following processing:

1. **PLOG Data**
   Processing begins with raw PLOG data. PLOG data is written by ADABAS as variable length records, with a PLOG file containing all transactions that occurred against a specific ADABAS for the time period for which the PLOG was active.

2. **ADASEL**
   The ADABAS utility ADASEL is used to decompress the PLOG records of interest. The execution of the ADASEL utility provides the data from PLOG transactions in a variable-length sequential file, with the ADASEL output either containing transaction records from one single file or all files, based on the parameters used with ADASEL.

   As an alternative to the use of ADASEL, the ADABAS utility ADACDC can also be used – please contact NatWorks for further details on the use of ADACDC over ADASEL.

3. **SORT**
   The next step in the processing is the execution of a system SORT which will sort the records into ascending sequence by File Number, ISN and Transaction Sequence.

4. **NATCDCSP**
   The NatWorks utility NATCDCSP is used to process the output of the SORT. NATCDCSP reads the variable-length sequential input file and selects the records which pertain to the files of interest. When a transactional record is selected, NATCDCSP converts the variable-length record into a fixed-length record, and it can optionally "drop" any fields which are not of interest to the user.

   NATCDCSP operates by receiving a NatQuery-generated parameter file, with this parameter file controlling how the fixed-length records should be created for each file, and also dictating what (if any) fields should be dropped from each file. While processing, NATCDCSP inserts a "File Header" record into the output at the beginning of transactions for a given file being processed and also inserts a "File Trailer" record after all transactions for a given file have been handled.

5. **Generated Processing Program(s)**
   The final step in the processing is the execution of a generated Natural program (a "traffic" program), with this program then calling generated Natural Subprograms.

   The Traffic Program operates by reading a record from the output of NATCDCSP, and the first time it does this it will encounter a "File Header" record. Based on the File

Number (FNR) found in this File Header record, the Traffic program CALLNATs a generated Natural Subprogram that was generated to specifically handle the processing of that specific File. The Natural Subprogram continues reading from the output of NATCDCSP and process the records found for the specific file until this Subprogram encounters the File Trailer record, at which time program control is returned to the Traffic Program. The Traffic Program then reads the next record, which will be another "File Header" record, at which point the Traffic Program will then branch to another generated Natural Subprogram that is specific to handling the transactions of this next file. This processing continues until all records are read from the NATCDCSP output.

Optionally, the generated Subprograms can deliver "Delta" images, they can deliver the First and Last images for each ISN found, or they can deliver all images (pass through all images – suitable for auditing).

The Subprograms will convert any and all "difficult" ADABAS data-types into ASCII equivalents, including properly handling Date, Time and Timestamp fields.

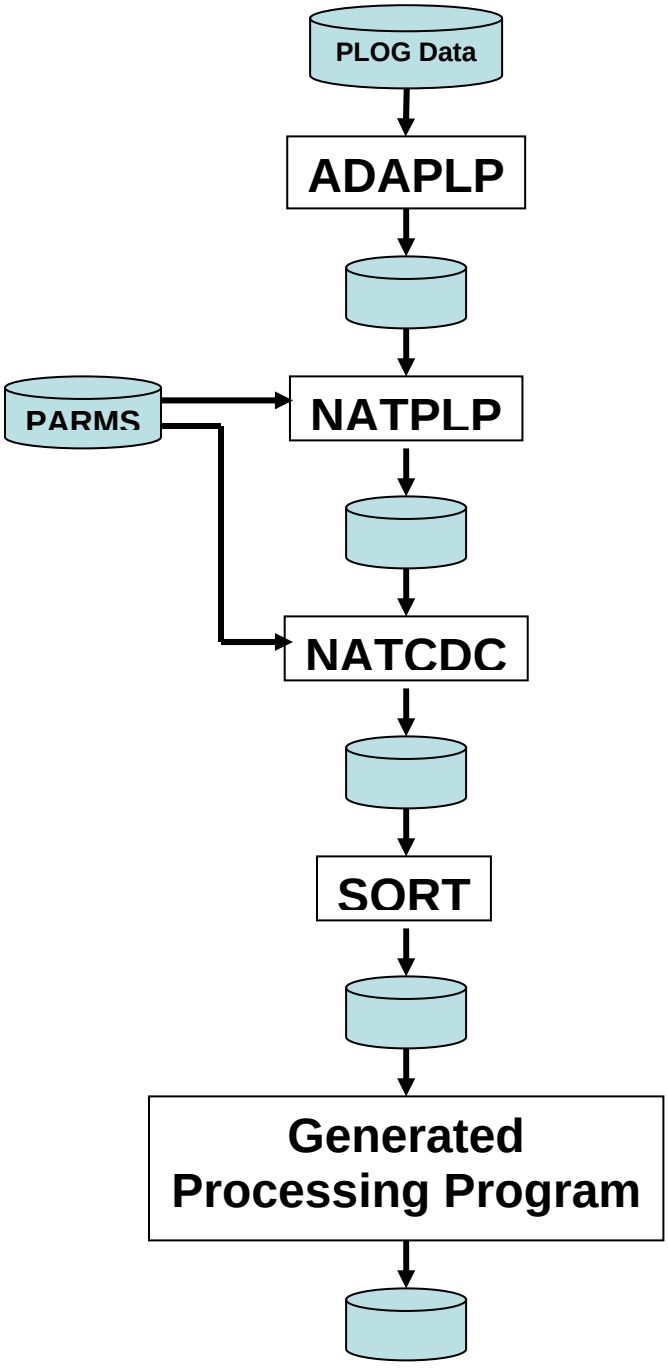
Finally, the Subprograms also produce numerous counts and totals in reports which details exactly what each of the Subprograms handled. The Traffic Program then produces overall totals.

6. **Splitter Program (optional)**
   Depending on how the delivered PLOG data will be handled, a need may arise to have utility programs available that will read through the single output file created by the Generated Processing Programs and then "split out" transactions from files into separate files – one for each file handled.

   NatWorks currently delivers two "flavors" of splitter programs: A program designed to operate in the Windows environment, and another designed to operate in the UNIX / Linux environment.

NATCDC and Open Systems Processing

The following graphic visually depicts the processing of NATCDCSP in UNIX, Linux and Windows environments. A description of this processing follows below.



Process Description for NATCDC on Open Systems Platforms

The above graphic depicts the following processing:

1. **PLOG Data**
   Processing begins with raw PLOG data. PLOG data is written by ADABAS as variable length records, with a PLOG file containing all transactions that occurred against a specific ADABAS for the time period for which the PLOG was active.

2. **ADAPLP**
   The ADABAS utility ADAPLP is used to decompress the PLOG records of interest. The execution of the ADAPLP utility essentially provides the data from PLOG transactions in a "report" format, with the ADAPLP output either containing transaction records from one single file or all files, based on the parameters used with ADAPLP.

3. **NATPLP**
   The NatWorks utility NATPLP is then used against the output of ADAPLP. NATPLP essentially parses through the ADAPLP "report" and converts the data into variable-length data records.

   NATPLP operates by receiving a NatQuery-generated parameter file, with this parameter file controlling which transaction records the user wishes to process and therefore which transactional records NATPLP will select and output.

4. **NATCDC**
   The NatWorks utility NATCDC is used to process the output of NATPLP. NATCDC reads the variable-length sequential input file and selects the records which pertain to the given file of interest (NATCDC is designed to process only a single file in a single pass). When a record is selected, NATCDC converts the variable-length record into a fixed-length record, and it can optionally "drop" any fields which are not of interest to the user.

   NATCDC operates by receiving a NatQuery-generated parameter file, with this parameter file controlling how the fixed-length record should be created, and also dictating what (if any) fields should be dropped.

5. **SORT**
   The next step in the processing is the execution of a system SORT which will sort the records into ascending sequence by ISN and Transaction Sequence.

6. **Post-Processing Program**
   The final step in the processing is the execution of a generated Natural program that is specific to the file being handled. Optionally, this generated program can deliver "Delta" images, it can deliver the First and Last images for each ISN found, or it can deliver all

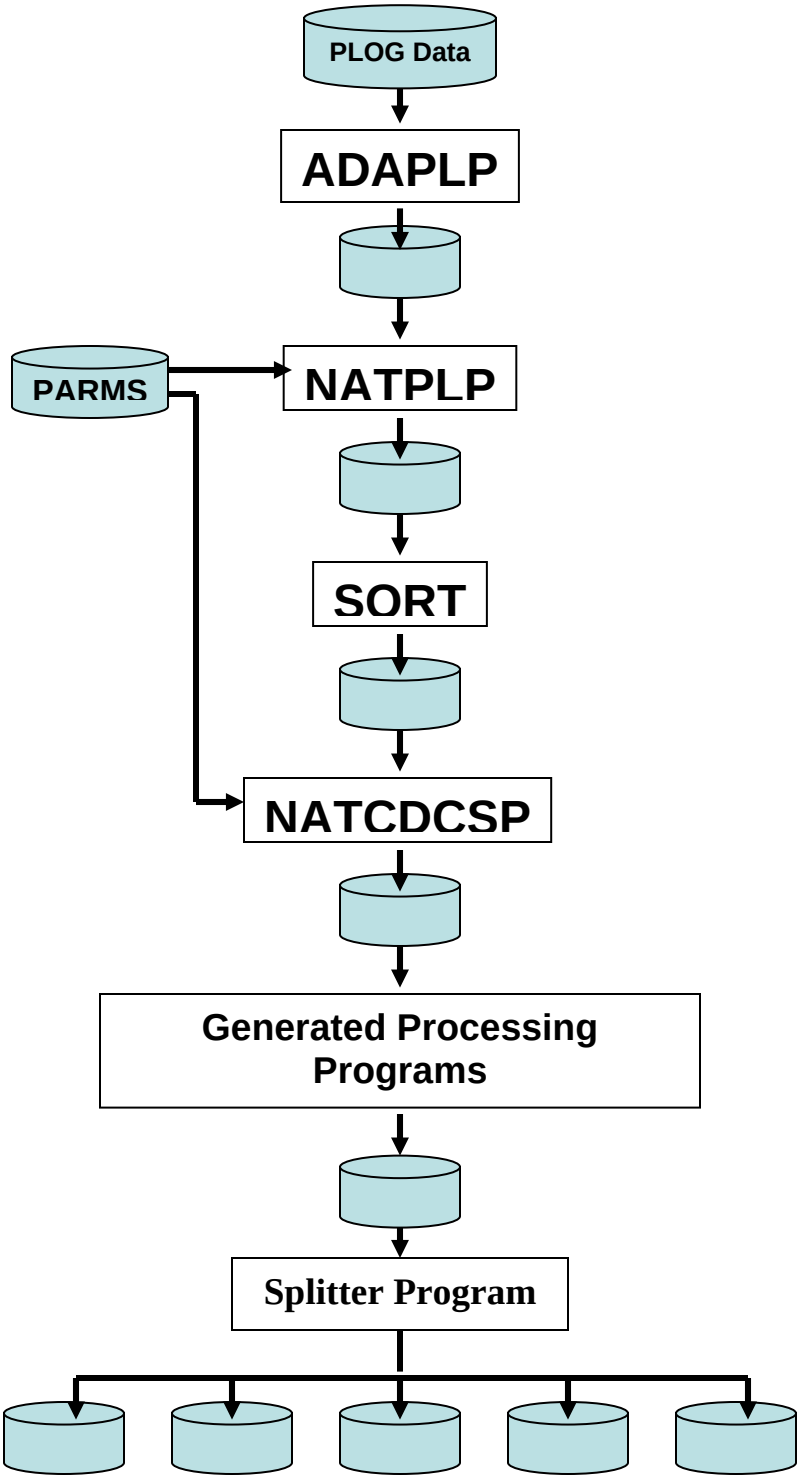
images (pass through all images – suitable for auditing).

The Post-Processing program will convert any and all “difficult” ADABAS data-types into ASCII equivalents, including properly handling Date, Time and Timestamp fields.

The post-processing program also produces numerous counts and totals in a report which details exactly what the post-processing encountered and what it output.

NATCDCSP and Open Systems Processing

The following graphic visually depicts the processing of NATCDCSP in UNIX, Linux and Windows environments. A description of this processing follows below.



Process Description for NATCDCSP on Open System Platforms

The above graphic depicts the following processing:

1. **PLOG Data**
   Processing begins with raw PLOG data. PLOG data is written by ADABAS as variable length records, with a PLOG file containing all transactions that occurred against a specific ADABAS for the time period for which the PLOG was active.

2. **ADAPLP**
   The ADABAS utility ADAPLP is used to decompress the PLOG records of interest. The execution of the ADAPLP utility essentially provides the data from PLOG transactions in a "report" format, with the ADAPLP output either containing transaction records from one single file or all files, based on the parameters used with ADAPLP.

3. **NATPLP**
   The NatWorks utility NATPLP is used against the output of ADAPLP. NATPLP essentially parses through the ADAPLP "report", selects transactional records specified by the user (through the input of Parameter Cards) and converts the data into variable-length data records.

4. **SORT**
   The next step in the processing is the execution of a system SORT which will sort the records into ascending sequence by File Number, ISN and Transaction Sequence.

5. **NATCDC**
   The NatWorks utility NATCDCSP is used to process the output of the SORT. NATCDCSP reads the variable-length sequential input file and selects the records which pertain to the files of interest. When a transactional record is selected, NATCDCSP converts the variable-length record into a fixed-length record, and it can optionally "drop" any fields which are not of interest to the user.

   NATCDCSP operates by receiving a NatQuery-generated parameter file, with this parameter file controlling how the fixed-length record should be created, and also dictating what (if any) fields should be dropped. While processing, NATCDCSP inserts a "File Header" record into the output at the beginning of transactions for a given file being processed and also inserts a "File Trailer" record after all transactions for a given file have been handled.

6. **Generated Processing Programs**
   The final step in the processing is the execution of a generated Natural program (a "traffic" program), with this program then calling generated Natural Subprograms.

The Traffic Program operates by reading a record from the output of NATCDCSP, and the first time it does this it will encounter a "File Header" record. Based on the File Number (FNR) found in this File Header record, the Traffic program CALLNATs a generated Natural Subprogram that was generated to specifically handle the processing of that specific File. The Natural Subprogram continues reading from the output of NATCDCSP and process the records found for the specific file until this Subprogram encounters the File Trailer record, at which time program control is returned to the Traffic Program. The Traffic Program then reads the next record, which will be another "File Header" record, at which point the Traffic Program will then branch to another generated Natural Subprogram that is specific to handling the transactions of this next file. This processing continues until all records are read from the NATCDCSP output.

Optionally, the generated Subprograms can deliver "Delta" images, they can deliver the First and Last images for each ISN found, or they can deliver all images (pass through all images – suitable for auditing).

The Subprograms will convert any and all "difficult" ADABAS data-types into ASCII equivalents, including properly handling Date, Time and Timestamp fields.

Finally, the Subprograms also produce numerous counts and totals in reports which details exactly what each of the Subprograms handled. The Traffic Program then produces overall totals.

7. **Splitter Program (optional)**
   Depending on how the delivered PLOG data will be handled, a need may arise to have utility programs available that will read through the single output file created by the Generated Processing Programs and then "split out" transactions from files into separate files – one for each file handled.

   NatWorks currently delivers two "flavors" of splitter programs: A program designed to operate in the Windows environment, and another designed to operate in the UNIX / Linux environment. A Splitter designed to operate in a mainframe environment can be delivered upon request.

Installation of NATCDC / NATCDCSP

Installation of NATCDC and / or NATCDCSP involves the introduction of the NATCDC, NATCDCSP programs and supporting modules into NATURAL in the server environment.

Subsequent to the installation of these NATURAL objects, NatQuery / NatCDC must be provided with a NatCDC License Key to enable NatCDC-related functions.

With NatCDC functions enabled, Configuration can occur that will then allow for the Generation of required PLOG processing programs.

Due to differences in how NatWorks handles the delivery of the NATCDC and NATCDCSP objects, the installation of these objects is different for Mainframes versus UNIX, Linux and Windows ("Open Systems"). Please refer to the appropriate section for detailed Installation information.

Available sections are:

- Installing into a Mainframe Environment
- Installing into an Open System Environment

Installation into a Mainframe Environment

For mainframe environments, NatWorks delivers two Natural Programs to support PLOG processing, with these programs being NATCDC and NATCDCSP. Both NATCDC and NATCDCSP are designed to work with the output of the ADABAS utility ADASEL. Support for the ADABAS utility ADACDC is also available through a special version of NATCDC, however NATCDCSP support for ADACDC is not currently offered (this will be supported in future versions however).

For Mainframe platforms, NATCDC and NATCDCSP are delivered in Natural Object Code form with no Natural Source Code provided.

When transporting Natural Object Code created in a mainframe environment outside of the mainframe environment (such as across the internet or across Workstations), a problem arises with the fact that this Object Code is EBCDIC-based binary written as variable-length records, and this format can (and will) become corrupted when it is moved through or across an ASCII environment (such as across the internet).

To remove this problem, subsequent to a NATCDC or NATCDCSP module being compiled into Natural Object Code in a mainframe environment, NatWorks unloads the Natural Object Code modules using the NATUNLD utility of Natural. The output of NATUNLD is then processed through a NatWorks-written Natural utility program (called CBIN2HEX), with this utility converting the EBCDIC binary variable-length records into fixed-length hexadecimal records. This fixed-length hexadecimal file, referred to as a "**Transport Module**", can then be safely moved outside of the mainframe environment as the hexadecimal representation is fully ASCII compatible. Once the **Transport Module** is placed onto the desired target mainframe environment, a second NatWorks-provided Natural utility program is run (called CHEX2BIN) against the Transport Module, with this utility converting the hexadecimal representation back into EBCDIC binary representation. This binary file can then be used as input to the Natural NATLOAD utility.

CHEX2BIN is provided in the FILES sub-directory of a NatQuery installation, with this being provided as Natural Source code. A copy of CBIN2HEX is available by request from NatWorks.

The typical installation of NATCDC and / or NATCDCSP into a mainframe environment involves the following four steps (individually detailed further below):

1. Locate or otherwise obtain the NATCDC and / or NATCDCSP **Transport Module**(s).

2. Moving the NATCDC and / or NATCDCSP **Transport Module**(s) into the mainframe environment.

3. Running the NatWorks hexadecimal-to-binary conversion utility called CHEX2BIN (provided with a NatQuery / NATCDC installation) to convert the **Transport Module**(s) into a NATLOAD-compatible Natural Object Module(s).

4. Loading the NATCDC and / or NATCDCSP Object Module(s) into Natural using the NATLOAD utility of Natural.

While each of the above steps can be handled manually, steps 2, 3 and 4 are fully automated into a single function of NatQuery.

To proceed with the installation of NATCDC and / or NATCDCSP, please refer to one of the following sections:

- Fully Automated Installation of NATCDC and / or NATCDCSP,
- Partially Automated Installation of NatCDC and / or NATCDCSP, or
- Manual Installation of NATCDC and / or NATCDCSP

Fully Automated Installation (Mainframes)

The successful completion of this approach will be dependent on the following:

- NatQuery should be properly configured to use *Direct FTP* against the mainframe environment. *Direct FTP* relates to NatQuery's ability to place Job Control Language (JCL) streams directly into JES (for MVS, OS/390 or Z/OS type systems) or POWER (for VSE systems) using FTP. For details on *Direct FTP* – please refer to the NatQuery Installation and Operations Manual.

- The user who will be performing the NATCDC or NATCDCSP install must be defined as a user to NatQuery and Natural (if Natural Security is installed).

- NatQuery should have the **Production Request Process** template configured for use.

- The User must have access to the NATLOAD utility of Natural

- A NATCDC and / or NATCDCSP Transport Module should be available, with these modules being available for authorized download from the NatWorks' website.

Assuming the above requirements are met, the following steps will install the core NATCDC / NATCDCSP module in a fully automated fashion:

1. **Locate / Obtain the Transport Module for NATCDC and / or NATCDCSP**
   While NATCDC and NATCDCSP software may have been provided to you via a CD, in almost all cases the best approach to obtaining the latest version of a NATCDC / NATCDCSP **Transport Module** is to download the required module from the NatWorks website (www.natworks-inc.com) using this link. Access to this page will require a NatWorks-supplied **User ID** and **Password**: If you do not have or have otherwise misplaced this information, you will need to contact NatWorks (see the section entitled Contacting NatWorks later in this manual).

   When downloaded from the website, a downloaded **Transport Module** will be named in one of three ways: CDC**XYZ**.zip, CDCA**XYZ**.zip or CDCSP**XYZ**.zip.

   For **Transport Modules** prefixed with "**CDC**", this prefix indicates that the module supports PLOG processing using the output of the ADABAS utility ADASEL. This type of module will support PLOG processing of transactions against a single ADABAS file in a single execution; the Natural name of such a module is typically "**NATCDC**".

   For **Transport Modules** prefixed with "**CDCA**", this prefix indicates that the module supports PLOG processing using the output of the ADABAS utility ADACDC. This type

of module will support PLOG processing of transactions against a single ADABAS file in a single execution; the Natural name of such a module is typically "**NATCDCA**".

For **Transport Modules** prefixed with "**CDCSP**", this prefix indicates that the module supports PLOG processing against the output of the ADABAS utility ADASEL. This type of module will support PLOG processing of transactions against multiple ADABAS files in a single execution; the Natural name of such a module is typically "**NATCDCSP**".

The "**XYZ**" portions of this naming equates to single-digit numeric values where "**X**" indicates the Major Version, "**Y**" indicates the Minor Version, and "**Z**" indicates the Maintenance release.

After downloading the appropriate **Transport Module**(s), a file will be present on the workstation that will have a name similar to "CDC**XYZ**.zip", CDCA**XYZ**.zip" or "CDCSP**XYZ**.zip". Make specific note of the directory into which the download occurred as this information will be needed in the subsequent step.

2. **"Unzip" the Downloaded Transport Module**
   If the Transport Module(s) was downloaded from the NatWorks Website, then these will need to be "unzipped" (decompressed) using a utility designed to handle this. This is a requirement because NatWorks zips these files prior to uploading them on the website to make the file smaller and also because zipping provides a level of security.

   Unzipping the downloaded file will then result in the creation of a file with a name similar to "CDC**XYZ**.hex", "CDCA**XYZ**.hex" or "CDCSP**XYZ**.hex".

   Most current Windows operating systems will unzip a zipped file by locating the file in Windows Explorer and double-clicking it. Alternatively, free versions of **Zip Reader**, an unzipping and decompression utility can be downloaded from PKWARE, located at www.pkware.com.

3. **Use Install / Update NatCDC Module Function**
   If you have NatQuery version 3.2.5 or higher installed, you may now use the **Install / Update NatCDC Module** function to process the NatCDC **Transport Module** into Natural on your mainframe server. On an empty NatQuery desktop, this function is invoked by clicking on the **Administer** drop-down menu**,** clicking on the **NatCDC / NATCDCSP** menu item, and then clicking on **Install / Update NatCDC Module** item**.**

   If you do not have NatQuery version 3.2.5, then it is strongly advised that you upgrade to the latest version of NatQuery prior to proceeding any further with the NATCDC, NATCDCA or NATCDCSP installation.

Detailed Help on using the **Install / Update NatCDC Module** function is available by clicking the **Help** button while in this function. If this function operates as intended, then the result should be a JCL stream being placed directly into the internal reader on the mainframe (JES or POWER), or otherwise being placed into pre-defined mainframe dataset. If the later occurred, then the steps necessary to submit the JCL to JES or POWER should now be taken.

When the JCL finishes executing, the output of this job should be scrutinized to insure that all steps ran properly. Essentially, the submitted JCL should use the **Transport Module** as input to the Natural program CHEX2BIN. The output of CHEX2BIN is a binary file, which can then be used as input to an execution of the NATLOAD utility of Natural.

If errors occur in executing this JCL, corrective action should be taken to correct the cause of the error and the JCL should then be rerun until the JCL finishes properly.

Partially Automated Installation (Mainframes)

The successful completion of this approach will be dependent on the following:

- NatQuery should be configured to use FTP against the mainframe environment, even if there is no intention (or even the possibility) of actually using this connectivity. Essentially, NatQuery will in many cases allow the use of functions that are dependent upon FTP if NatQuery is told that FTP is available. In this situation, NatQuery will perform the required generation, but prior to physically invoking FTP NatQuery will prompt the user with a Message Box asking if it is okay to initiate the FTP. When this prompting occurs, NatQuery has internally generated (and then saved into text files) any object that is needed for the specific FTP action. By answering 'No' to NatQuery's prompt to initiate FTP: NatQuery will bypass the FTP completely and will then effectively end the processing of the current function, but leaves the generated objects that would have been used with the FTP protocol (JCL, Natural Programs, Parameter Files, Etc.). With these objects generated, it is then usually a simple (but manual) process that will allow the object(s) to be moved into the mainframe environment where they can then be manually submitted / executed.

- NatQuery should have the **Production Request Process** template configured for use.

- The user who will be performing the NATCDC or NATCDCSP install must be defined as a user to NatQuery and Natural (if Natural Security is installed).

- The User must have access to the NATLOAD utility of Natural

- A NATCDC and / or NATCDCSP Transport Module should be available, with these modules being available for authorized download from the NatWorks' website.

Assuming the above requirements are met, the following steps will install the core NATCDC / NATCDCSP module in a Partially Automated fashion: The successful completion of this approach will take advantage of NatQuery's ability to automatically generate the process required to introduce a NatCDC or NATCDCSP into Natural, with manual intervention then handling the introduction of that process into the mainframe environment and the execution of that process. With this approach, there is absolutely no dependency that NatQuery be able to successfully connect to the mainframe via FTP; he just needs to "think" he can.

The following steps will install the core NATCDC / NATCDCSP module in a partially automated fashion:

1. **Locate / Obtain the Transport Module for NATCDC and / or NATCDCSP**
   While NATCDC and NATCDCSP software may have been provided to you via a CD, in

almost all cases the best approach to obtaining the latest version of a NATCDC / NATCDCSP **Transport Module** is to download the required module from the NatWorks website (www.natworks-inc.com) using this link. Access to this page will require a NatWorks-supplied **User ID** and **Password**: If you do not have or have otherwise misplaced this information, you will need to contact NatWorks (see the section entitled Contacting NatWorks later in this manual).

When downloaded from the website, a downloaded **Transport Module** will be named in one of three ways: CDC**XYZ**.zip, CDCA**XYZ**.zip or CDCSP**XYZ**.zip.

For **Transport Modules** prefixed with "**CDC**", this prefix indicates that the module supports PLOG processing using the output of the ADABAS utility ADASEL. This type of module will support PLOG processing of transactions against a single ADABAS file in a single execution; the Natural name of such a module is typically "**NATCDC**".

For **Transport Modules** prefixed with "**CDCA**", this prefix indicates that the module supports PLOG processing using the output of the ADABAS utility ADACDC. This type of module will support PLOG processing of transactions against a single ADABAS file in a single execution; the Natural name of such a module is typically "**NATCDCA**".

For **Transport Modules** prefixed with "**CDCSP**", this prefix indicates that the module supports PLOG processing against the output of the ADABAS utility ADASEL. This type of module will support PLOG processing of transactions against multiple ADABAS files in a single execution; the Natural name of such a module is typically "**NATCDCSP**".

The "**XYZ**" portions of this naming equates to single-digit numeric values where "**X**" indicates the Major Version, "**Y**" indicates the Minor Version, and "**Z**" indicates the Maintenance release.

After downloading the appropriate **Transport Module**(s), a file will be present on the workstation that will have a name similar to "CDC**XYZ**.zip", CDCA**XYZ**.zip" or "CDCSP**XYZ**.zip". Make specific note of the directory into which the download occurred as this information will be needed in the subsequent step.

2. **"Unzip" the Downloaded Transport Module**
   If the Transport Module(s) was downloaded from the NatWorks Website, then these will need to be "unzipped" (decompressed) using a utility designed to handle this. This is a requirement because NatWorks zips these files prior to uploading them on the website to make the file smaller and also because zipping provides a level of security.

   Unzipping the downloaded file will then result in the creation of a file with a name similar to "CDC**XYZ**.hex", "CDCA**XYZ**.hex" or "CDCSP**XYZ**.hex".

Most current Windows operating systems will unzip a zipped file by locating the file in Windows Explorer and double-clicking it. Alternatively, free versions of **Zip Reader**, an unzipping and decompression utility can be downloaded from PKWARE, located at www.pkware.com.

3. **Use Install / Update NatCDC Module Function**
   If you have NatQuery version 3.2.5 or higher installed, you may now use the **Install / Update NatCDC Module** function to process the NatCDC **Transport Module** into a single JCL stream that will, when manually moved to the mainframe and then manually executed, introduce a NATCDC or NATCDCSP Object Module into Natural on your mainframe server. On an empty NatQuery desktop, this function is invoked by clicking on the **Administer** drop-down menu**,** clicking on **NatCDC / NATCDCSP** menu item, and then clicking on **Install / Update NatCDC Module** item**.**

   If you do not have NatQuery version 3.2.5, then it is strongly advised that you upgrade to the latest version of NatQuery prior to proceeding any further with the NATCDC, NATCDCA or NATCDCSP installation.

   Detailed Help on using the **Install / Update NatCDC Module** function is available by clicking the **Help** button while in this function.

   At some point in the generation process, NatQuery will prompt the user with a Message Box that looks similar to the following:

   When this prompt occurs, the user should click the “No” button.

4. As a result of performing the above steps, a file will be created in the Environment Path that will be named as “*user-id*.JCL”, where *user-id* is the value of the User-ID provided to NatQuery through the NatQuery Configuration function.

   This file will be the JCL needed to handle the loading of the NATCDC / NATCDCSP Transport Module into Natural on the mainframe.

   The current Environment Path of NatQuery can be seen on the NatQuery desktop, and

can also be seen and set through the NatQuery Configuration function.

5. Using whatever steps are necessary, the file "*user-id*.JCL" should now be moved in some fashion into the mainframe environment.

   Once in the mainframe environment, whatever steps are required to execute the JCL stream should now be taken.

6. When the JCL has finished executing, the administrator should scrutinize the output to insure that that:

   - The conversion program CHEX2BIN ran successfully
   - The NATLOAD utility ran successfully

7. If any errors are seen in the JCL, then corrective action should be taken against the JCL with the JCL being re-submitted until the JCL runs correctly.

8. The result of the above steps should load a NATCDC or NATCDCSP module into Natural in the mainframe environment.

Manual Installation (Mainframes)

The following steps will allow for the manual install of the core NATCDC / NATCDCSP module(s):

1. **Locate / Obtain the Transport Module for NATCDC and / or NATCDCSP**
   While NATCDC and NATCDCSP software may have been provided to you via a CD, in almost all cases the best approach to obtaining the latest version of a NATCDC / NATCDCSP Transport Module is to download the required module from the NatWorks website (www.natworks-inc.com) using this link. Access to this page will require a NatWorks-supplied **User ID** and **Password**: If you do not have this information you will need to contact NatWorks.

   When downloaded from the website, a downloaded Transport Module will be named as CDC**XYZ**.zip, CDCA**XYZ**.zip or CDCSP**XYZ**.zip. The "**XYZ**" portions of this naming equate to single-digit numeric values where "**X**" indicates the Major Version, "**Y**" indicates the Minor Version, and "**Z**" indicates the Maintenance release.

   For Transport Modules prefixed with "**CDC**", this prefix indicates that the module supports PLOG processing using the output of the ADABAS utility ADASEL. This type of module will support PLOG processing of transactions against a single ADABAS file in a single execution; the internal name of such a module is typically "**NATCDC**".

   For Transport Modules prefixed with "**CDCA**", this prefix indicates that the module supports PLOG processing using the output of the ADABAS utility ADACDC. This type of module will support PLOG processing of transactions against a single ADABAS file in a single execution; the internal name of such a module is typically "**NATCDCA**".

   For Transport Modules prefixed with "**CDCSP**", this prefix indicates that the module supports PLOG processing against the output of the ADABAS utility ADASEL. This type of module will support PLOG processing of transactions against multiple ADABAS files in a single execution; the internal name of such a module is typically "**NATCDCSP**".

   After downloading the appropriate module, a file will be present on the workstation that will have a name similar to "CDC**XYZ**.zip", CDCA**XYZ**.zip" or "CDCSP**XYZ**.zip".

2. **"Unzip" the Downloaded Transport Module**
   Subsequent to being downloaded; the Transport Module(s) then have to unzipped using a utility such as WINZIP, PKZIP or PKUNZIP.

   Unzipping the downloaded file will then result in the creation of a file with a name

similar to “CDC**XYZ**.hex”, “CDCA**XYZ**.hex” or “CDCSP**XYZ**.hex”.

3. **Locate / Obtain CHEX2BIN Natural Program**
   In the **Files** sub-directory of the NatQuery install directory (this path will typically be C:\Program Files\NatQuery\Files\), locate a file called CHEX2BIN.txt; alternatively you can download this file from the NatWorks website (www.natworks-inc.com) by using this link.

   If the file is downloaded, then it will be downloaded with the name of CHEX2BIN.zip, and this file should be unzipped to create CHEX2BIN.txt.

   The file CHEX2BIN.txt is a simple Natural Program provided in Natural Source Code form that is designed to reconstruct a NATLOAD-compatible file from a NatWorks Transport Module.

4. **Allocate Mainframe Datasets**
   Allocate two datasets on your mainframe, defined as follows:

   File #1
   Suggested Name: ***user-id*.TRANSFER.BIN**
   RECFM=FB
   LRECL=252
   BLKSIZE=25200

   File #2
   Suggested Name: ***user-id*.TRANSFER.HEX**
   RECFM=FB
   LRECL=126
   BLKSIZE=28350

5. Using FTP or a similar utility, move the file created step #2 above (example: “CDCXYZ.hex”, into **File #2**. The “.hex” file will be a pure ASCII file, *and should not be moved as Binary*.

   NOTE:
   On VSE, and possibly on Z/OS as well depending on how FTP is configured, and assuming that you are using FTP to move this file into the mainframe, you should issue the following FTP commands to prior to FTPing this file:

   QUOTE SITE LRECL 126
   QUOTE SITE BLKSIZE 28350
   QUOTE SITE RECFM FB

6. Move the Natural program **CHEX2BIN,** handled in step 3 above, into the target natural environment on your server and STOW this program.

7. Build a batch Natural JCL stream that will execute the program **CHEX2BIN**, with Work File 1 (CMWKF01) of this JCL pointing at **File #1**, and Work File 2 (CMWKF02) pointing at **File #2** and then execute this program.

   If this program runs properly, it should create a binary file in File #1, with this file being a Natural Object Module in NATLOAD format.

8. Run the Natural NATLOAD utility against **File #1**. The parameters for this execution of NATLOAD should be:

   NATLOAD ALL * REPLACE NEWLIB *target-library-name*

9. If any errors are seen in the JCL, then corrective action should be taken against the JCL with the JCL being re-submitted until the JCL runs correctly.

10. The result of the above steps should load a NATCDC or NATCDCSP module into Natural in the mainframe environment.

Installation into an Open Systems Environment

For UNIX / Linux and Windows environments, NatWorks delivers three Natural Programs to support PLOG processing, with these programs being NATPLP, NATCDC and NATCDCSP. An optional Natural-based SORT program is also made available, with this program being named NATSORT.

In all cases, the NATPLP module is required in UNIX, Linux and Windows environments as it is this module's responsibility to convert the report-oriented output of NATPLP into a data file that can then be consumed by NATCDC or NATCDCSP. Depending on the processing needs of the individual user site: NATCDC, NATCDCSP or both modules will be required. The NATSORT module is only needed when a binary-capable SORT program is not available in the Natural / ADABAS environment – however due to the limitations of using a Natural-based SORT program – it is STRONGLY advised that any UNIX, Linux or Windows based ADABAS system that is looking to process PLOGS have available a Production-Strength SORT utility (such as CO-SORT, found on the web at www.cosort.com).

The typical installation of NATCDC and / or NATCDCSP into a UNIX, Linux or Windows environment involves the following three steps (individually detailed further below):

1. Locate or otherwise obtain the NATCDC / NATCDCSP SYSOBJH file.

2. Move the NATCDC / NATCDCSP SYSOBJH file into the server environment.

3. Run the SYSOBJH utility of Natural, using the Load option, using as input the NATCDC / NATCDCSP SYSOBJH file.

While each of the above steps can be handled manually, steps 2 and 3 are fully automated into a single function of NatQuery.

To proceed with the installation of NATCDC and / or NATCDCSP, please refer to one of the following sections:

- Fully Automated Loading the NATCDC / NATCDCSP

- Manual Loading of NATCDC / NATCDCSP

Once the NATCDC / NATCDCSP modules have been loaded into NATURAL in the server environment, the administrator can proceed to provide NatQuery with a NATCDC License Key.

Fully Automated Loading (Open Systems)

Installation of NATPLP, NATCDC, NATCDCSP or NATSORT is a straightforward process as all these program objects are provided in a format compatible with the Natural Utility SYSOBJH.

The successful completion of this approach will be dependent on the following:

- NatQuery should be properly configured to use *Direct FTP* or *Direct Copy* against the server environment. *Direct FTP* relates to NatQuery's ability to place Scripts and Script-dependent objects directly onto the server via FTP, where this Script will then be executed via Remote Execution. Direct Copy relates to NatQuery's ability to place Script streams and dependent objects directly onto a server via Network Copy operations, where the Script will also be executed via Remote Execution. For details on *Direct FTP* and *Direct Copy* – please refer to the NatQuery Installation and Operations Manual.

- The user who will be performing the NATCDC or NATCDCSP install must be defined as a user to NatQuery and Natural (if Natural Security is installed).

- NatQuery should have the **Production Request Process** template configured for use.

- The User must have access to the **SYSOBJH** utility of Natural.

- A NATCDC and / or NATCDCSP SYSOBJH Portable file should be available, with this module being available for authorized download from the NatWorks' website.

To install these modules, please perform the following steps:

1. **Locate / Obtain the SYSOBJH module for NATCDC / NATCDCSP**
   While NATCDC / NATCDCSP software may have been provided to you via a CD, in almost all cases the best approach is to obtain the latest version of NATCDC / NATCDCSP by downloading the latest SYSOBJH file from the NatWorks website (www.natworks-inc.com) using this link. Access to this page will require a NatWorks supplied **User ID** and **Password**: If you do not have or have misplaced this information, you will need to contact NatWorks (see the section entitled Contacting NatWorks later in this manual).

   When downloaded from the website, the SYSOBJH module will be named in a similar fashion to NATCDC_SYSOBJH.SAG. This single file then contains the Object Code for modules NATCDC, NATCDCSP and NATPLP, as well as the Source and Object Code

for the NATSORT module.

2. **Use Install / Update NatCDC Module Function**
   If you have NatQuery version 3.2.5 or higher installed, you may now use the **Install / Update NatCDC Module** function to process the SYSOBJH Portable file into a single Script that will, when executed, introduce a NATCDC or NATCDCSP Object Module into Natural on your UNIX, Linux or Windows NATURAL server.

   On an empty NatQuery desktop, this function is invoked by clicking on the **Administer** drop-down menu**,** clicking on **NatCDC / NATCDCSP** menu item, and then clicking on **Install / Update NatCDC Module** item**.**

   If you do not have NatQuery version 3.2.5, then it is strongly advised that you upgrade to the latest version of NatQuery prior to proceeding any further with the NATCDC, NATCDCA or NATCDCSP installation.

   Detailed Help on using the **Install / Update NatCDC Module** function is available by clicking the **Help** button while in this function.

   Subsequent to clicking **OK** on the **Install / Update NATCDC Module** function, NatQuery will prompt for the movement of the SYSOBJH Portable file onto the server, followed by a prompt to move the Script and supporting files to the server.

   For all server platforms, the following files should be generated and moved to the server:

   *user-id*_CMOBJIN.TXT,
   *user-id*_CMSYNIN.TXT, and
   *user-id*_CDC.SAG

   For UNIX and Linux servers, a file called "*user-id*.SH" will also be moved; for Windows servers this file will be called "*user-id*.BAT".

   If NatQuery is properly configured for *Direct FTP* or *Direct Copy*, then NatQuery should have prompted the user for permission to remotely execute the Script on the server platform.

   The user should now check the server to see if the Script executed properly, with this execution having invoked SYSOBJH which then should have loaded NATCDC, NATCDCSP, NATPLP and NATSORT into Natural on the server. If difficulties are found with the execution of this script then corrective action should be taken with the Script or the environment such that the script executes successfully.

Manual Installation (Open Systems)

The following steps will allow for the manual install of NATCDC, NATCDCSP, NATPLP and NATSORT into UNIX, Linux or Windows server environments.

The successful completion of this approach will be dependent on the following:

- The user who will be performing the installation of the NATCDC - related modules must be defined as a user to NATURAL (if Natural Security is installed).
- The User must have access to the **SYSOBJH** utility of Natural
- A NATCDC and / or NATCDCSP SYSOBJH Portable file must be available, with this module being available for authorized download from the NatWorks' website.
- In some Natural environments, it will be necessary to have a permanent assignment in NATPARM for Work File 3, such as "/wrkfile3.txt". This will avoid a NAT1500 error that can occur if there is no value set for Work File 3.

To install these modules, please perform the following steps:

1. **Locate / Obtain the SYSOBJH module for NATCDC / NATCDCSP**
   While NATCDC / NATCDCSP software may have been provided to you via a CD, in almost all cases the best approach is to obtain the latest version of NATCDC / NATCDCSP by downloading the latest SYSOBJH file from the NatWorks website (www.natworks-inc.com) using this link. Access to this page will require a NatWorks-supplied **User ID** and **Password**: If you do not have or have otherwise misplaced this information, you will need to contact NatWorks (see the section entitled Contacting NatWorks later in this manual).

   When downloaded from the website, the SYSOBJH module will be named in a similar fashion to NATCDC_SYSOBJH.SAG. This single file contains the Object Code for modules NATCDC, NATCDCSP and NATPLP, as well as the Source and Object Code for the NATSORT module.

2. **Move SYSOBJH module onto Server**
   Using manual FTP, a copy operation across a common network or other means, the SYSOBJH module should be moved onto the UNIX, Linux, or Windows server.

3. **Invoke the SYSOBJH Utility on the Server**
   The **SYSOBJH** utility of NATURAL is invoked while in NATURAL by entering a command line of "SYSOBJH".

Due to the differences in interacting with SYSOBJH in character-based environments versus graphical environments, please refer to the appropriate section below.

Procedure for SYSOBJH on UNIX and Linux Platforms
If the server is a UNIX or Linux system, then the SYSOBJH command has been invoked in the character-based NATURAL environment. If this is the case, then please use the following commands:

a. On the **SYSOBJH** Main Menu, select the **Load** option and click Enter.

b. On the initial **Load Wizard** screen, select **Load objects from Natural work file(s)** and click Enter.

c. On the **Load Wizard, Options** screen, select the **Portable work file** option, then enter the path and name of the Load File (**Work file**) that was moved onto the server in step 2 above, and then select **Set additional options**.

d. On the **Load Options** screen, insure that the **Write report** option is selected, and then enter the path and name of the **Report file** that will be created as a result of the execution of **SYSOBJH**.

   Under the **Report Options**, it is recommended that the **Replace all** option should be selected over the default value of; **Do not replace**.

   Now click Enter.

e. On the **Load Wizard, Options** screen, click Enter.

f. On the **Load Wizard, Parameters** screen, de-select **Do not use parameters**, and select **Use global parameters** and **Set global parameters**. Now click Enter.

g. On the **Load Parameters** screen, under the **New Value** column, enter the **Library** name into which the SYSOBJH objects should be loaded into. Now click Enter.

h. On the **Load Wizard, Parameters** screen, click Enter.

i. On the **Load Wizard, Select Load Type** screen, insure that the **Load all objects from the work file** is selected, and click Enter.

j. On the **Load Wizard** screen, click Enter.

Assuming that no errors resulted from performing the above, the user should now be notified that the function completed successfully. A **Display Load Report** screen should then display the load results. If any errors occurred during the above procedure, appropriate corrective action should be taken that result in the proper loading of the SYSOBJH modules.

Procedure for SYSOBJH on Windows Systems
If the server is a Windows platform, then it is likely that the SYSOBJH command has been invoked in the GUI NATURAL environment. If this is the case, then the following commands:

a. On the SYSOBJH Main Menu, click the **Load** option.

b. On the initial **Load Wizard** window, select **Load objects from Natural work file(s)** and click **Next**.

c. On the next **Load Wizard, Options** window, select the **Portable work file** option, then Browse to or enter the path and name of the Load File (**Work file**) that was moved onto the server in step 2 above.

   Select **Use additional options**, and then click **Set**.

d. On the **SYSOBJH – Additional Load Options** window, **General** tab, insure that the **Write report** option is selected, then Browse to or enter the path and name of the **Report file** that will be created as a result of the execution of SYSOBJH.

   On the **Special** tab, select the **Replace all** option, then click **OK**.

e. On the **Load Wizard** window, click **Next**.

f. On the next **Load Wizard** window, select **Use global parameters** and click **Set.**

g. On the **Load Parameters** window, **General** tab, under the **New Value** column, enter the **Library** name into which the SYSOBJH objects should be loaded into and then click **OK**.

h. On the **Load Wizard** window, click **Next**.

i. On the next **Load Wizard** window**,** insure that the **Load all objects from the work file** is selected, and then click **Next**.

j. On the next **Load Wizard** screen, click **Next**.

Assuming that no errors resulted from performing the above, the user should now be notified that the function completed successfully. A **Display Load Report** screen should then display the load results. If any errors occurred during the above procedure, appropriate corrective action should be taken such that the proper loading of the SYSOBJH modules occur.

NATCDC License Key

In order for NatQuery to provide access to NATCDC-specific functions, it is required that NatQuery be given a valid license keys for NATCDC. NatQuery will use this License Key to allow access to NatQuery functions designed for use with NATCDC, and the core NATCDC processing module will subsequently use this key as well.

To provide NatQuery with a NATCDC License Key, perform the following:

1. **Start NatQuery**
   If NatQuery is already started, insure that NatQuery is on an "open" desktop (I.E. no query open, and no Administrative windows open).

2. **Invoke NatQuery Configuration**
   The NatQuery Configuration function allows several facets of information to be changed, and the type of information handled includes the processing and capture of License Keys for both NatQuery and NATCDC.

   To invoke the NatQuery Configuration function, perform the following: Click on **Administer** and then click on **NatQuery Configuration**.

3. **Enter NATCDC License Key**
   When the **NatQuery Configuration** window is invoked, it presents a tab control that allows access to the varying types of information that the function handles. By default, the **User Identification** tab is presented.

   For the entry of License Keys, click on the **License Keys** tab.

   In the appropriate text boxes, enter the **NATCDC License Key** as provided to you by NatWorks or your NatWorks partner company.

4. **Close the NatQuery Configuration function**
   After entering a valid **NatCDC License Key**, the user can click the **OK** button to close the **NatQuery Configuration** window.

Assuming that the NATCDC / NATCDCSP modules have been installed into NATURAL on the server environment as described in a previous section, and NatQuery has been provided with a NatCDC License Key as described above, the Administrator may proceed to Configuring JCL / Script as outlined in the following section.

Configuring JCL / Script

Once the NATCDC / NATCDCSP modules have been loaded into the appropriate Natural library and NatQuery has been given NatCDC License Keys, the last step prior to Generation involves the configuration of the JCL (mainframes) or Script / bat files (Open Systems) that will physically execute a NATCDC or a NATCDCSP process.

JCL / Script for both NATCDC and NATCDCSP is based on templates; and due to the differences between the functioning of NATCDC versus NATCDCSP, each module has it own JCL or Script template.

Since executing NATURAL in batch with JCL in mainframe environments is quite a bit different then executing NATURAL in batch in a UNIX, Linux, or Windows environments, please refer to one of the following two sections for configuration information that is specific to your server platform:

- JCL Configuration (Mainframes)
- Script Configuration (UNIX. Linux and Windows)

JCL Configuration (Mainframes)

Job Control Language (JCL) for NATCDC and NATCDCSP processing is handled through NatQuery's JCL / Script Information function. This function is accessible for an Administrative installation of NatQuery by clicking on the **Administer** drop-down menu, then clicking **Environment Configuration**, then clicking **Server Connection Configuration**, and then clicking **JCL / Script Information**.

When NatQuery / NatCDC generated JCL is required to introduce one or more NATURAL programs into a Natural environment, this processing is done by executing an "E" (Edit) command in a batch stream, then listing the program (referenced by dynamic-substitution variable(s) which will be placed into the NATURAL editor in batch), followed by the ".E" command to end the editor.

For information relating to configuring the JCL template for NATCDC or NATCDCSP, please refer to one of the following two sections:

- NATCDC JCL Configuration
- NATCDCSP JCL Configuration

NATCDC JCL Configuration (Mainframes)

For NATCDC processing, the JCL template called **Production NATCDC Process** is used.

For information on how to configure a mainframe JCL template for NATCDCSP, please refer to the section entitled NATCDCSP JCL Configuration.

This section is split out into the following topics:

- Discussion of NATCDC JCL Processing, and
- Configuring the NATCDC JCL Template.

Once the NATCDC / NATCDCSP module(s) have been installed into NATURAL, NatQuery has been given a valid NATCDC License Key, and a NATCDC (or NATCDCSP) JCL template has been configured: The administrator can then proceed to Generation of PLOG processing.

Discussion of NATCDC JCL Processing (Mainframes)

As an overview, and as can be seen in the graphic entitled NATCDC and Mainframe Processing, a single NATCDCSP process for a given ADABAS PLOG source file involves the use of seven possible files / datasets, as described below:

1. **PLOG Data (ADASEL-OUTPUT)**
   This file will be the variable-length output for an execution of ADASEL, for discussion purposes, this file will be referred to as **ADASEL-OUTPUT**.

   This file will be used as the primary input to a NATCDC execution.

2. **Parameter File (CDC-PARMS)**
   This file-specific parameter file (generated by NatQuery) will be used by NATCDC so that it can properly understand how to transform a variable-length ADASEL output file into a fixed-length format. For discussion purposes, this file will be referred to as **CDC-PARMS**.

   This file will be used as the secondary input to a NATCDC execution; this file will be created as result of Generation.

3. **Interim File #1 (CDC-I-DSNAME1)**
   This variable-length temporary file will contain the fixed-length data output of a NATCDC execution, written by NATCDC as variable-length records. For discussion purposes, this file will be referred to as **CDC-I-DSNAME1**.

   This file will be used as input into a System SORT program.

4. **Interim File #2 (CDC-I-DSNAME2)**
   This variable-length temporary file will result from running a System SORT against **Interim File #1**. For discussion purposes, this file will be referred to as **CDC-I-DSNAME2**.

   This file will be used as input to a generated NATURAL post-NATCDC processing program.

5. **Sort Work Area (CDC-TEMPSORT - optional)**
   Depending upon how the system SORT program operates, a temporary file may need to be allocated that will be utilized as a work area to the SORT program. For discussion purposes, this file will be referred to as **CDC-TEMPSORT**.

   The **CDC-TEMPSORT** file name is typically named using a dynamic substitution

variable prefix with a "hard-coded" suffix, as this is a temporary file.

6. **Final Output (CDC-F-DSNAME)**
   This fixed-length file is the data that results from a complete NATCDC process, written as fixed-length records with fixed-length fields. For discussion purposes, this file will be referred to as **CDC-F-DSNAME**.

7. **Final-Totals (CDC-TOTALS - optional)**
   This fixed-length file is an optional file that can be created at a user's request. If requested, this file will contain a single line that will detail the number of records read, along with the number of Stores, Updates and Deletes that is contained in **CDC-F-DSNAME** described above.

As part of a normal NatQuery installation, NatWorks delivers an example Script template that will serve as a basis for developing a customer-specific NATCDC JCL process, with this template being called **Production NATCDC Process**. This example JCL template must be customized in order to be utilized within a customer's unique environment.

The Production NATCDC Process JCL template will typically contain 4 steps, one of which is optional depending upon the operating system of the server.

- **STEP 1**
  This step performs any required allocation of datasets needed for the NATCDC process, usually needed only for VSE systems using VSAM. To assist in the proper definition of these files and their attributes, NatQuery makes use of Dynamic Substitution Variables that will provide for file-specific values for things like File Names, Logical Record Lengths, Allocation Units, Primary and Secondary number of Allocation Units, Block Size, etc.

  For an OS/390 (MVS) system this step is not typically required due to the ability to allocate required files as they are referenced in individual steps.

  For VSE systems, and due to the *suggested* use of VSAM files, the required files; CDC-I-DSNAME1, CDC-I-DSNAME2, and CDC-F-DSNAME) should be allocated in this step. If required, a temporary SORT work area (CDC-TEMPSORT) may also need to be defined. If DYNAM is used over VSAM, then this step is optional.

- **STEP 2**
  This step will execute a batch Natural environment to execute the core NATCDC object module. NATCDC then converts the variable-length input file (ADASEL-OUTPUT) into fixed-length output, written by NATCDC as Variable Length records.

To allow NATCDC to perform the appropriate mapping of variable-length records into a fixed-length output records, NATCDC will input a generated parameter file specific to the given source file (CDC-PARMS, read as Work File 1). Using the user-specifications contained in CDC-PARMS, NATCDC will read ADASEL-OUTPUT (read as Work File 2), create a fixed number of occurrences for each recurring field (padding or truncating as needed), and will additionally "drop" any unwanted fields for the final output record.

NATCDC will output CDC-I-DSNAME1 (written to Work File 3).

- **STEP 3 (required)**
  This step executes a system SORT for the purpose of sorting the transactions against the specific ADABAS source file into ascending sequence by ISN and Sequence-Number, with these two fields being located in the first 12 bytes of the CDC-I-DSNAME1 output (positions 5-16 inclusive of the 4-byte variable-record length).

  The SORT will take as input the file created in STEP 2 (CDC-I-DSNAME1), and will create a sorted output file (CDC-I-DSNAME2).

- **STEP 4 (required)**
  This step will execute a batch Natural environment to execute a NatQuery-generated post-NATCDC processing program. This program will properly interpret the sorted output of NATCDC into the transactions requested (Logical Last or Delta, Logical First and Last, or All), and will perform any required data transformations / translations, with the final output being fully ASCII compatible with fixed-length records with fixed-length fields.

  This step will take as input the file created in STEP 3 (CDC-I-DSNAME2 – read as Work File 1), and will create the final output file according to user-supplied specifications (CDC-F-DSNAME – written as Work File 2).

  Optionally, the NatQuery-generated post-NATCDC processing program will create CDC-TOTALS, written as Work File 3).

Configuring the NATCDC JCL Template (Mainframes)

To configure that NATCDC JCL template, perform the following steps:

1. **Start NatQuery**
   If NatQuery is already started, then insure that NatQuery is on an "open" desktop (I.E. no queries open and no Administrative windows are open.

2. **Invoke Administer JCL / Script function**
   The Administer JCL / Script function provides NatQuery with the ability of handling JCL templates. To invoke this function, click on **Administer**, click on **Environment Configuration**, click on **Server Connection Configuration**, and then click on **JCL / Script Information**.

   This will invoke the **Administer JCL / Script** window.

3. **Select Production NATCDC Process Template**
   In the upper right hand corner of the **Administer JCL / Script** window, there is a selection-box that controls which JCL template will be presented for editing or modification.

   To invoke the JCL template for NATCDC, click on this selection-box and then select the **Production NATCDC Process** template.

   When initially selected, there will be no Production version for the NATCDC template, and the large text box in the middle of the window will be populated with a message to this effect. In response to this message, it is *suggested* to use the installation-provided example JCL template as a base. This is accomplished by clicking on the **Copy From Example Template** button while the Production NATCDC Process template is selected.

   This action will cause NatQuery to copy the example installation-provided NATCDC JCL template into the central text box (this text will then become the **Production NATCDC Process** template when it is saved).

4. **Modify the NATCDC JCL / Template**
   At this point, the Administrator should customize the Production NATCDC Process JCL template according to site requirements.

   The types of modifications needed will typically include modifying:

   - The Job Card(s),

- The two batch NATURAL steps so that they properly reference the name of the appropriate batch NATURAL nucleus and pass NATURAL the appropriate parameters,

- Possibly modifying the SORT step to properly utilize the available mainframe SORT utility, and

- Disk Name references.

When making changes and modifications to the **Production NATCDC Process** template, the designated NATCDC Administrator must be aware of the fact that the template being built is a "generic" template, a template that provides the proper steps and syntax to perform a "generic" NATCDC process. This "generic" template is made unique to a specific NATCDC PLOG process through the use of "dynamic substitution" variables, which are designated variables that will be replaced by appropriate values during the NATCDC process generation.

While there are many dynamic substitution variables that NatQuery can handle, there is specific dynamic substitution variables designated for use with NATCDC. For a discussion on the NATCDC-specific dynamic substitution variables, please refer to the section entitled Dynamic Substitution Variable Reference.

5. **Save Changes**
   When the **Production NATCDC Process** JCL template has been changed to properly reflect the customer's environment, this JCL template should be saved by clicking the **Save** button.

6. **Close the Administer JCL / Script Window**
   Once the **Production NATCDC Process** JCL template has been saved, the administrator can then close the **Administer JCL / Script** window by clicking the **OK** button

At this point in time, the JCL just created will not be used until Generation has been performed and the JCL template is actually executed. As with all JCL, it may take several iterations of changes to the **Production NATCDC Process** template to achieve a template that works as required.

NATCDCSP JCL Configuration (Mainframes)

For NATCDCDSP processing, the JCL template called **Production NATCDC SP Process** is used.

For information on how to configure a mainframe JCL template for NATCDC, please refer to the section entitled NATCDC JCL Configuration.

This section is split out into the following topics:

- Discussion of NATCDCSP Processing, and
- Configuring the NATCDCSP JCL Template.

Once the NATCDC / NATCDCSP module(s) have been installed into NATURAL, NatQuery has been given a valid NATCDC License Key, and a NATCDCSO (or NATCDC) JCL template has been configured: The administrator can then proceed to Generation of PLOG processes.

Discussion of NATCDCSP JCL Processing (Mainframes)

As an overview, and as can be seen in the graphic entitled NATCDCSP and Mainframe Processing, a single NATCDCSP process involves the use of six possible files / datasets, as described below:

1. **ADASEL Output File (ADASEL-OUTPUT)**
   The variable-length output of an execution of ADASEL. For discussion purposes, this file will be referred to as **ADASEL-OUTPUT**.

   This file will be used as the primary input to a system SORT execution.

2. **Parameter File (CDC-PARMS)**
   The parameter file (generated by NatQuery) that instructs NATCDCSP on how to transform a variable-length ADASEL output records into a fixed-length format. For discussion purposes, this file will be referred to as **CDC-PARMS**.

   This is a generated file that will be used as the input to a NATCDCSP execution; **CDC-PARMS** are generated by NatQuery as a result of user-specifications.

3. **Interim File #1 (CDC-I-DSNAME1)**
   The variable-length temporary file that will contain the fixed-length data output of a NATCDC execution, written by NATCDC as variable-length records. For discussion purposes, this file will be referred to as **CDC-I-DSNAME1**.

   This file will be used as output of a System SORT program.

4. **Interim File #2 (CDC-I-DSNAME2)**
   The variable-length temporary file that will result from executing NATCDCSP against CDC-I-DSNAME1. For discussion purposes, this file will be referred to as **CDC-I-DSNAME2**.

   This file will be used as input to a generated NATURAL post-NATCDCSP processing program that will call a series of generated NATURAL sub-programs (one generated sub-program for each requested file).

5. **Sort Work Area (CDC-TEMPSORT - optional)**
   Depending upon how the system SORT program operates, a temporary file may need to be allocated that will be utilized as a work area to the SORT program. For discussion purposes, this file will be referred to as **CDC-TEMPSORT**.

6. **Final Output (CDC-F-DSNAME)**
   This file is the data that results from a complete NATCDCSP process, written by a series

of generated NATURAL sub-programs as variable-length records with fixed-length fields (all records written for a given FNR will all be the same fixed-length). For discussion purposes, this file will be referred to as **CDC-F-DSNAME**.

As part of a normal NatQuery installation, NatWorks delivers a JCL template that will serve as a basis for developing a customer-specific NATCDCSP JCL process, with this template being called **Production NATCDC SP Process**. While NatQuery installs this template and makes it available as an example it will need to be customized in order to be useful within a customer's specific environment.

Within the installation-provided NATCDCSP processing template, there are four basic "job steps", some of which may be optional depending on the specific operating system of the server. Each of these steps is discussed below:

- **STEP 1 (optional)**
  This step performs any required allocation of datasets needed for the NATCDCSP process, usually needed only for VSE systems using VSAM. To assist in the proper definition of these files and their attributes, NatQuery makes use of Dynamic Substitution Variables that will provide for file-specific values for things like File Names, Logical Record Lengths, Allocation Units, Primary and Secondary number of Allocation Units, Block Size, etc.

  For an OS/390 (MVS) system this step is not typically required due to the ability to allocate required files as they are referenced in individual steps.

  For VSE systems however, and due to the *suggested* use of VSAM files, the required files CDC-I-DSNAME1, CDC-I-DSNAME2, CDC-F-DSNAME) should be allocated in this step. If required, a temporary SORT work area (CDC-TEMPSORT) may also need to be defined. If DYNAM is used over VSAM, then this step is optional.

- **STEP 2 (required)**
  This step executes a system SORT for the purpose of sorting the transactions into FNR / ISN sequence while retaining the original sequence of the sorted records. The SORT will sort on File Number (FNR, position 7-8 counting the 4-byte inclusive record length) and also on Internal Sequence Number (ISN, position 33-36 counting the 4-byte inclusive record length).

  The SORT will take as input the file created from an execution of ADASEL (ADASEL-OUTPUT), and will create a sorted output file (CDC-I-DSNAME1).

- **STEP 3 (required)**
  This step will execute a batch Natural environment to execute the core NATCDCSP

object module. NATCDCSP then converts variable-length input file (CDC-I-DSNAME1) into fixed-length output for each file being processed, but written by NATCDCSP as Variable Length records across all files.

To allow NATCDCSP to perform the appropriate mapping of transactions from individual ADABAS source files, NATCDCSP will input a generated parameter file that contains information on each file to be processed (CDC-PARMS, read as Work File 1).

Using the user-specifications contained in CDC-PARMS, NATCDCSP will read CDC-I-DSNAME2 (read as Work File 2), create a fixed number of occurrences for each recurring field (padding or truncating as needed) in a given source file, and will additionally "drop" any unwanted fields for the final output record according to user-specifications.

NATCDCSP will output CDC-I-DSNAME2 (written to Work File 3).

- **STEP 4 (required)**
  This step will execute a batch Natural environment to execute a NatQuery-generated post-NATCDCSP processing program. This "Traffic Program" will in turn call individual generated NATURAL sub-programs, with each of these sub-programs properly handling the transactions requested (Logical Last or Delta, Logical First and Last, or All) for a specific requested file.

  This step will take as input the file created by the NATCDCSP module in STEP 3 (CDC-I-DSNAME2 – read as Work File 1), and will create the final output file according to user-supplied specifications (CDC-F-DSNAME – written as Work File 2).

Configuring the NATCDCSP JCL Template (Mainframes)

To configure that NATCDCSP JCL template, perform the following steps:

1. **Start NatQuery**
   If NatQuery is already started, then insure that NatQuery is on an "open" desktop (I.E. no queries open and no Administrative form are open.

2. **Invoke Administer JCL / Script function**
   The **Administer JCL / Script** function provides NatQuery with the ability of handling JCL templates. To invoke this function, perform the following:

   Click on **Administer**, then click on **Environment Configuration**, then click on **Server Connection Configuration**, and then click on **JCL / Script Information**.

   This will invoke the **Administer JCL / Script** window.

3. **Select NATCDC JCL Template**
   In the upper right hand corner of the **Administer JCL / Script** window there is a selection-box that controls the JCL template that will be presented.

   To invoke the JCL template for NATCDCSP, click on this selection-box and then select the **Production NATCDC SP Process** template.

   When initially selected, there will be no production version for the NATCDCSP template, and the text box in the middle of the window will be populated with a message to this effect. To resolve this and to use the installation-provided JCL template as a base, click on the **Copy From Example Template** button. This action will cause NatQuery to copy the example installation-provided NATCDCSP JCL template into the central text box (this text will then become the **Production NATCDC SP Process** template when it is saved).

4. **Modify the NATCDCSP JCL / Template**
   At this point, the Administrator should customize the Production NATCDC Process JCL template according to site requirements.

   The types of modifications needed will typically include:

   - The Job Card(s),

- The two batch NATURAL steps so that they properly reference the name of the appropriate batch NATURAL nucleus and pass NATURAL the appropriate parameters,

- Possibly modifying the SORT step to properly utilize the available mainframe SORT utility, and

- Disk Name references, etc.

When making changes and modifications to the **Production NATCDC SP Process** template, the designated NATCDCSP Administrator must be aware of the fact that the template being built is a "generic" template, a template that provides the proper steps and syntax to perform a "generic" NATCDCSP process. This "generic" template is made unique to a specific NATCDCSP PLOG process through the use of "dynamic substitution" variables, which are designated variables that will be replaced by appropriate values during the NATCDCSP process generation.

While there are many dynamic substitution variables that NatQuery can handle, there are specific dynamic substitution variables designated for use with NATCDCSP. For a discussion on the NATCDC-specific dynamic substitution variables, please refer to the section entitled Dynamic Substitution Variable Reference.

5. **Save Changes to Production NATCDCSP Process Template**
   When the **Production NATCDC SP Process** JCL template has been changed to properly reflect the customer's environment, this JCL template should be saved by clicking the **Save** button.

6. **Close the Administer JCL / Script Window**
   Once the **Production NATCDC SP Process** JCL template has been saved, the administrator can then close the **Administer FTP JCL / Script** window by clicking the **OK** button

At this point in time, the JCL just created will not be used until Generation has been performed and the JCL resulting from generation is actually executed. As with all JCL, it may take several iterations of changes to the **Production NATCDC SP Process** template to achieve a template that works as required.

Script Configuration (Open Systems)

Scripts for NATCDC and NATCDCSP processes are handled through NatQuery's **Administer JCL / Script Information** function. This function is accessible for an Administrative installation of NatQuery from an open desktop by clicking on the **Administer** drop-down menu, then clicking **Environment Configuration**, then clicking **Server Connection Configuration**, and then clicking **JCL / Script Information**.

When dealing with the NATURAL batch execution against UNIX, Linux and Windows, NatQuery follows Software AG's recommendations on handling batch processing, meaning that there will typically be multiple files generated to support a single batch execution. Generally, there will be a:

- **CMSYNIN File**
  This file is intended to be used for data intended to be read by Natural INPUT statements.

- **CMOBJIN File**
  This file is intended to be used to contain specific NATURAL commands

- **Shell Script or Bat File**
  For UNIX and Linux systems, Shell Scripts should be generated; for Windows systems, a Bat File should be generated. Both a Shell Script and a Bat File will reference the CMOBJIN and CMSYSNIN files mentioned above, and may additionally reference other ancillary files (such as generated files containing generated NATURAL programs).

To simplify the administration of Script / Bat File templates, NatQuery handles all of these various batch components within a single file, with each individual component designated by "tags".

A typical NATCDC or NATCDCSP Script template will therefore usually contain the following tags:

- ***!*CMSYNIN INPUT FOLLOWS THIS LINE**
  All lines following this tag, up to the first line encountered that contains a "*!*" in the first three characters, will become a separate file that represents the CMSYNIN input file.

- ***!*CMOBJIN INPUT FOLLOWS THIS LINE**
  All lines following this tag, up to the first line encountered that contains a "*!*" in the first three characters will become a separate file that represents the CMOBJIN input file.

- ***!*SCRIPT FOLLOWS THIS LINE**
  All lines following this tag, up to the first line encountered that contains a "*!*" in the first three characters will become the Shell Script or Bat File that will physically execute the NATCDC / NATCDCSP process.

- ***!*!*!*!***
  This tag is used to represent the end of a SCRIPT / Bat File. Usually, a dynamic substitution variable called &&USER-EXTRACT-PROGRAM follows this tag

- ***!*STEP**
  This tag is typically used exclusively in Windows environment where ".bat" files are used for execution. In this environment, a "*!*STEP" command will have the effect of hitting the ENTER key.

- ***REP>**
  Any line that has this tag as a 5-character prefix will be repeated by the NatQuery Script processing. If this tag is repeated across several lines; all lines with this prefix will be repeated as a block of lines.

  Note: This type of structure will only be utilized by NATCDCSP.

When NatQuery / NatCDC generated Scripts are required to introduce one or more NATURAL programs into a server's NATURAL environment, this processing is done through a NatWorks module called NQYPNT05, which is introduced into NATURAL as a result of installing and configuring NatQuery. NQYPNT05 makes use of a user-exit (NATURAL Sub-Program) provided by Software AG named USR0080N. Generated Scripts will therefore call NQYPNT05 as appropriate to introduce generated NATURAL programs or sub-programs into NATURAL.

For information relating to configuring the Script template for NATCDC or NATCDCSP, please refer to one of the following two sections:

- NATCDC Script Configuration
- NATCDCSP Script Configuration

NATCDC Script Configuration (Open Systems)

For NATCDC processing, the Script template called **Production NATCDC Process** is used.

For information on how to configure an Open Systems Script template for NATCDCSP, please refer to the section entitled NATCDCSP Script Configuration.

This section is split out into the following topics:

- Discussion of NATCDC Script Processing, and
- Configuring the NATCDC Script Template.

Once the NATCDC / NATCDCSP module(s) have been installed into NATURAL, NatQuery has been given a valid NATCDC License Key, and a NATCDC (or NATCDCSP) JCL template has been configured: The administrator can then proceed to Generation of PLOG processing.

Discussion of NATCDC Script Processing (Open Systems)

As an overview, and as can be seen in the graphic entitled NATCDC and UNIX, Linux, Windows Processing, a single NATCDC process for a given ADABAS PLOG source file involves the use of eight possible files / datasets, as described below:

1. **ADAPLP Output File (ADASEL-OUTPUT)**
   This file is the output of an execution of ADAPLP. For discussion purposes, this file will be referred to as **ADASEL-OUTPUT** (even though the output is from ADAPLP).

   This file will be used as the primary input to a NATPLP, a NatWorks-provided NATCDC pre-processor.

2. **Parameter File (CDC-PARMS)**
   This is a file-specific parameter file (generated by NatQuery) that instructs the NatWorks utility program NATPLP on which transactions should be selected from ADAPLP output, based on the transaction's File Number (FNR).

   This parameter file is also utilized by NATCDC so that it can properly understand how to transform a variable-length NATPLP output record into a fixed-length format. For discussion purposes, this file will be referred to as **CDC-PARMS**.

3. **Interim File #0 (CDC-I-DSNAME0)**
   This is a variable-length temporary file that will contain the output of the NatWorks utility NATPLP. This output is then used as input to the NATCDC module.

   Both **CDC-I-DSNAME0** and **CDC-TEMPSORT** (described further below) are values that should be named in Script using dynamic substitution variable prefix values with a "hard-coded" suffix, as they are both temporary files that have no full replacement dynamic substitution values (in contrast to the other files described in this section).

4. **Interim File #1 (CDC-I-DSNAME1)**
   A variable-length temporary file that will contain the fixed-length data output of a NATCDC execution, written by NATCDC as variable-length records. For discussion purposes, this file will be referred to as **CDC-I-DSNAME1**.

   This file will then be used as input into a System SORT program.

5. **Interim File #2 (CDC-I-DSNAME2)**
   A variable-length temporary file that will result from running a System SORT against **Interim File #1**. For discussion purposes, this file will be referred to as **CDC-I-DSNAME2**.

This file will be used as input to a generated NATURAL post-NATCDC processing program.

6. **Sort Work Area (CDC-TEMPSORT - optional)**
   Depending upon how the system SORT program operates, a temporary file may need to be allocated that will be utilized as a work area to the SORT program. For discussion purposes, this file will be referred to as CDC-TEMPSORT.

   Both **CDC-TEMPSORT** and **CDC-I-DSNAME0** (described above) are values that should be named in Script using dynamic substitution variable prefix values with a "hard-coded" suffix, as they are both temporary files that have no full replacement dynamic substitution values (in contrast to the other files described in this section).

7. **Final Output (CDC-F-DSNAME)**
   This fixed-length file is the data that results from a complete NATCDC process, written as fixed-length records with fixed-length fields. This file is created by a NatQuery-generated NATURAL program that will read **CDC-I-DSNAME2** and performing the user-specified processing against this input.

   For discussion purposes, this file will be referred to as **CDC-F-DSNAME**.

8. **Final-Totals (CDC-TOTALS - optional)**
   This fixed-length file is an optional file that can be created at a user's request. If requested, this file will contain a single line that will detail the number of records read, along with the number of Stores, Updates and Deletes that are contained in **CDC-F-DSNAME**.

As part of a normal NatQuery installation, NatWorks delivers a Script template that will serve as a basis for developing a customer-specific NATCDC Script process, with this template being called **Production NATCDC Process**. While NatQuery installs this template and makes it available as an example, in order to utilize this Script template in a customer's environment this Script template must be customized.

Within the installation-provided NATCDC processing template, there will be references to four files (CMOBJIN, CMSYNIN, the Shell Script or Bat File, and a generated program). The portion that references the Shell Script or Bat File (I.E. all lines following the tag "***!*SCRIPT FOLLOWS THIS LINE**" up to the next line encountered that contains "*!*" as the first three characters), will contain five basic "job steps", some of which may be optional depending on network architecture. Each of these steps is discussed below:

- **STEP 1**
  This step executes the NatWorks utility program NATPLP followed by an execution of NATCDC in a batch NATURAL environment.

  NATPLP will use as input the output of ADAPLP (referred to as **ADASEL-OUTPUT** – read as Work File 4) and will also input the generated parameter file (**CDC-PARMS** – read as Work File 1).

  NATPLP will then create an output file (**CDC-I-DSNAME0** – written as Work File 2), which will then be used as input to an execution of NATCDC.

  NATCDC will then read **CDC-I-DSNAME0** (read as Work File 2), will also read **CDC-PARMS** (read as Work File 1), and will create **CDC-I-DSNAME1** (written as Work File 3).

- **STEP 2**
  This step executes a system SORT for the purpose of sorting the transactions against the specific ADABAS source file into ascending sequence by ISN and Sequence-Number, with these two fields being located in the first 12 bytes of the **CDC-I-DSNAME1** output (positions 5-16 inclusive of the 4-byte variable-record length).

  The SORT will take as input the file created in STEP 1 (**CDC-I-DSNAME1**), and will create a sorted output file (**CDC-I-DSNAME2**).

- **STEP 3**
  This step will introduce the generated NATURAL post-NATCDC processing program into NATURAL on the server in preparation for execution in the next step.

  To introduce this generated NATURAL program into NATURAL, the NatWorks utility program NQYPNT05 is used.

- **STEP 4**
  This step will execute the newly-introduced generated NATURAL post-NATCDC processing program. This program will read the output of the SORT (CDC-I-DSNAME2), and will create the final output (CDC-F-DSNAME) according to the user's processing requirements (Logical Last / Delta, Logical First and Last, or All).

- **STEP5 (optional)**
  If the method of integration is being done via automated Network Copy operations and the server is a UNIX or Linux platform, then this step is required to convert the UNIX or Linux text file(s) so that they have a Carriage-Return / Line Feed combination at the end of each line, making them compatible with Windows / DOS applications and platforms.

  This step is not required when FTP is being used, since FTP automatically takes care of this conversion.

Configuring the NATCDC Script Template (Open Systems)

To configure the NATCDC Script template, perform the following steps:

1. **Start NatQuery**
   If NatQuery is already started, then insure that NatQuery is on an "open" desktop (I.E. no queries open and no Administrative windows are open.

2. **Invoke Administer FTP JCL / Script function**
   The **Administer JCL / Script** function provides NatQuery with the ability of handling Script templates. To invoke this function, perform the following from an open desktop:

   Click on **Administer**, then click on **Environment Configuration**, then click on **Server Connection Configuration**, and then click on **JCL / Script Information**.

   This will invoke the **Administer JCL / Script** window.

3. **Select NATCDC JCL / Template**
   In the upper right hand corner of the **Administer JCL / Script** window there is a selection-box that controls the Script template that will be presented.

   To invoke the Script template for NATCDC, click on this selection-box and then select the **Production NATCDC Process** template.

   When initially selected, there will be no production version for the NATCDC template, and the text box in the middle of the window will be populated with a message to this effect. To resolve this, and to use the installation-provided Script template as a base, click on the **Copy From Example Template** button. This action will cause NatQuery to copy the example installation-provided NATCDC Script template into the central text box (this text will then become the **Production NATCDC Process** template when it is saved).

4. **Modify the NATCDC Script Template**
   At this point, the Administrator should customize the **Production NATCDC Process** Script template according to site requirements.

   The types of modifications needed will typically include:

   - Path to Environment Variables (referenced as HIQUAL),
   - The three batch NATURAL steps so that they properly reference the name of the appropriate batch NATURAL nucleus and pass NATURAL the appropriate

parameters, and

- Possibly modifying the SORT step to properly utilize the available server SORT utility.

When making changes and modifications to the **Production NATCDC Process** template, the designated NATCDC Administrator must be aware of the fact that the template being built is a "generic" template, a template that provides the proper steps and syntax to perform a "generic" NATCDC process. This "generic" template is made unique to a specific NATCDC PLOG process through the use of "dynamic substitution" variables, which are designated variables that will be replaced by appropriate values during the NATCDC process generation.

While there are many dynamic substitution variables that NatQuery can handle, there is specific dynamic substitution variables designated for use with NATCDC. For a discussion on the NATCDC-specific dynamic substitution variables, please refer to the section entitled Dynamic Substitution Variable Reference.

5. **Save Changes to Production NATCDC Process Template**
   When the Production NATCDC Process JCL template has been changed to properly reflect the customer's environment, this JCL template should be saved by clicking the **Save** button.

6. **Close the Administer JCL / Script Window**
   Once the **Production NATCDC Process** JCL template has been saved, the administrator can then close the **Administer JCL / Script** window by clicking the **OK** button

At this point in time, the Script just created will not be used until Generation has been performed and the Script resulting from generation is actually executed. As with all Script, it may take several iterations of changes to the **Production NATCDC Process** template to achieve a template that works as required.

NATCDCSP Script Configuration (Open Systems)

For NATCDCSP processing, the Script template called **Production NATCDC SP Process** is used.

For information on how to configure a mainframe Script template for NATCDC, please refer to the section entitled NATCDC Script Configuration.

This section is split out into the following topics:

- Discussion of NATCDCSP Script Processing, and
- Configuring the NATCDCSP Script Template.

Once the NATCDC / NATCDCSP module(s) have been installed into NATURAL, NatQuery has been given a valid NATCDC License Key, and a NATCDC (or NATCDCSP) JCL template has been configured: The administrator can then proceed to Generation of PLOG processing.

Discussion of NATCDCSP Script Processing (Open Systems)

As an overview, and as can be seen in the graphic entitled NATCDCSP and UNIX, Linux and Windows, a single NATCDCSP process involves the use of six possible files / datasets, as described below:

1. **ADAPLP Output File (ADASEL-OUTPUT)**
   The output produced by an execution of ADAPLP. For discussion purposes, this file will be referred to as **ADASEL-OUTPUT** (even though the output is from ADAPLP).

   This file will be used as the primary input to the NatWorks-provided NATPLP pre-NATCDCSP processing program.

2. **Parameter File (CDC-PARMS)**
   A parameter file (generated by NatQuery) that instructs the NatWorks-provided NATPLP pre-processing program on which transactions should be selected from ADAPLP output, based on the transaction's File Number (FNR).

   This parameter file is also utilized by NATCDCSP so that it can properly understand how to transform variable-length NATPLP output records into a fixed-length record for each file that is requested to be processed (written as variable-length records). For discussion purposes, this file will be referred to as **CDC-PARMS**.

3. **Interim File #0 (CDC-I-DSNAME0)**
   A variable-length temporary file that will contain the output of the NatWorks-provided NATPLP pre-processing program. This output is then used as input to a system SORT program.

   Both **CDC-I-DSNAME0** and **CDC-TEMPSORT** (described further below) are values that should be named in Script using dynamic substitution variable prefix values with a "hard-coded" suffix, as they are both temporary files that have no full replacement dynamic substitution values (in contrast to the other files described in this section).

4. **Interim File #1 (CDC-I-DSNAME1)**
   A variable-length temporary file that will contain the output of a system SORT program, which is then used as input to an execution of NATCDCSP. For discussion purposes, this file will be referred to as **CDC-I-DSNAME1**.

   This file will be used as the primary input to the NATCDCSP module.

5. **Interim File #2 (CDC-I-DSNAME2)**
   A variable-length temporary file that will result from executing NATCDCSP against CDC-I-DSNAME1. For discussion purposes, this file will be referred to as **CDC-I-**

DSNAME2.

This file will be used as input to a generated NATURAL post-NATCDCSP processing program which will call a series of generated NATURAL sub-programs (one generated sub-program for each requested file).

6. **Sort Work Area (CDC-TEMPSORT - optional)**
   Depending upon how the system SORT program operates, a temporary file may need to be allocated that will be utilized as a work area to the SORT program. For discussion purposes, this file will be referred to as **CDC-TEMPSORT**.

7. **Final Output (CDC-F-DSNAME)**
   This file is the data that results from a complete NATCDCSP process, written by a series of generated NATURAL sub-programs as variable-length records with fixed-length fields (all records written for a given FNR will all be the same fixed-length). For discussion purposes, this file will be referred to as **CDC-F-DSNAME**.

As part of a normal NatQuery installation, NatWorks delivers a Script template that will serve as a basis for developing a customer-specific NATCDCSP Script process, with this template being called **Production NATCDC SP Process**. While NatQuery installs this template and makes it available as an example, in order to utilize this Script template in a customer's environment this Script template must be customized.

Within the installation-provided NATCDCSP processing template, there are five basic "job steps", some of which may be optional depending on the specific operating system of the server. Each of these steps is discussed below:

- **STEP 1**
  This step executes the NatWorks utility program NATPLP.

  NATPLP will use as input the output of ADAPLP (referred to as **ADASEL-OUTPUT** – read as Work File 4) and will also input the generated parameter file (**CDC-PARMS** – read as Work File 1).

  NATPLP will then create an output file (**CDC-I-DSNAME0** – written as Work File 2), which will then be used as input to an execution of a system SORT.

- **STEP 2**
  This step executes a system SORT for the purpose of sorting the transactions into FNR / ISN sequence while retaining the original sequence of the input file. The SORT will sort on File Number (FNR, position 7-8 counting the 4-byte inclusive record length) and also on Internal Sequence Number (ISN, position 33-36 counting the 4-byte inclusive record

length).

The SORT will take as input the file created from an execution of NATPLP (ADASEL-OUTPUT), and will create a sorted output file (CDC-I-DSNAME1).

- **STEP 3**
  This step will execute a batch Natural environment to execute the core NATCDCSP object module. NATCDCSP then converts variable-length input file (CDC-I-DSNAME1) into fixed-length output for each file being processed, written by NATCDCSP as Variable Length records across all files.

  To allow NATCDCSP to perform the appropriate mapping of transactions from individual ADABAS source files, NATCDCSP will input a generated parameter file that contains information on each file to be processed (CDC-PARMS, read as Work File 1).

  Using the user-specifications contained in CDC-PARMS, NATCDCSP will read CDC-I-DSNAME2 (read as Work File 2), create a fixed number of occurrences for each recurring field (padding or truncating as needed) within a given source file, and will additionally "drop" any unwanted fields from each file for the final output record, according to user-specifications.

  NATCDCSP will output CDC-I-DSNAME2 (written to Work File 3).

- **STEP 4**
  This step will execute a batch Natural environment to execute a NatQuery-generated post-NATCDCSP processing program. This "Traffic Program" will in turn call individual generated NATURAL sub-programs, with each of these sub-programs properly handling the transactions requested (Logical Last or Delta, Logical First and Last, or All) for a specific requested file.

  This step will take as input the file created by the NATCDCSP module in STEP 3 (CDC-I-DSNAME2 – read as Work File 1), and will create the final output file according to user-supplied specifications (CDC-F-DSNAME – written as Work File 2).

- **STEP5 (optional)**
  If the method of integration is being done via automated Network Copy operations and the server is a UNIX or Linux platform, then this step is required to convert the UNIX or Linux text file(s) so that they have a Carriage-Return / Line Feed combination at the end of each line, making them compatible with Windows / DOS applications and platforms.

  This step is not required when FTP is being used, since FTP automatically takes care of

this conversion.

Configuring the NATCDCSP Script Template (Open Systems)

To configure that NATCDCSP Script template, perform the following steps:

1. **Start NatQuery**
   If NatQuery is already started, then insure that NatQuery is on an "open" desktop (I.E. no queries open and no Administrative windows are open.

2. **Invoke Administer JCL / Script function**
   The Administer JCL / Script function provides NatQuery with the ability of handling JCL / Script templates. To invoke this function, perform the following:

   Click on **Administer**, then click on **Environment Configuration**, then click on **Server Connection Configuration**, and then click on **JCL / Script Information**.

   This will invoke the **Administer JCL / Script** window.

3. **Select NATCDCSP JCL / Script Template**
   In the upper right hand corner of the **Administer JCL / Script** window there is a selection-box that controls the Script template that will be presented.

   To invoke the Script template for NATCDCSP, click on this selection-box and then select the **Production NATCDC SP Process** template.

   When initially selected, there will be no production version for the NATCDCSP template, and the text box in the middle of the window will be populated with a message to this effect. To resolve this and use the installation-provided Script template as a base, click on the **Copy From Example Template** button. This action will cause NatQuery to copy the example installation-provided NATCDCSP Script template into the central text box (this text will then become the **Production NATCDC SP Process** template when it is saved).

4. **Modify the NATCDCSP Script Template**
   At this point, the Administrator should customize the Production NATCDCSP Script template according to site requirements.

   The types of modifications needed will typically include modifying:

   - Path to Environment Variables (referenced as HIQUAL),

- The three batch NATURAL steps so that they properly reference the name of the appropriate batch NATURAL nucleus and pass NATURAL the appropriate parameters, and

- Possibly modifying the SORT step to properly utilize the available server SORT utility.

When making changes and modifications to the **Production NATCDC Process** template, the designated NATCDCSP Administrator must be aware of the fact that the template being built is a "generic" template, a template that provides the proper steps and syntax to perform a "generic" NATCDCSP process. This "generic" template is made unique to a specific NATCDCSP PLOG process through the use of "dynamic substitution" variables, which are designated variables that will be replaced by appropriate values during the NATCDCSP process generation.

While there are many dynamic substitution variables that NatQuery can handle, there is specific dynamic substitution variables designated for use with NATCDCSP. For a discussion on the NATCDCSP-specific dynamic substitution variables, please refer to the section entitled Dynamic Substitution Variable Reference.

5. **Save Changes to Production NATCDCSP Process Template**
   When the Production NATCDCSP Process Script template has been changed to properly reflect the customer's environment, this Script template should be saved by clicking the **Save** button.

6. **Close the Administer FTP JCL / Script Window**
   Once the **Production NATCDC SP Process** Script template has been saved, the administrator can then close the **Administer JCL / Script** window by clicking the **OK** button

At this point in time, the Script just created will not be used until Generation has been performed and the Script resulting from generation is actually executed. As with all Script, it may take several iterations of changes to the **Production NATCDC SP Process** template to achieve a template that works as required.

Generation

Once NATCDC and / or NATCDCSP is installed into NATURAL, and the **Production NATCDC Process** and / or the **Production NATCDC SP Process** JCL / Script is configured, all further NATCDC / NATCDCSP interaction will occur using NatQuery to generate required objects against provided DDMs.

In order to utilize the NATCDC / NATCDCSP generation functions of NatQuery, it is a requirement that NatQuery be installed and configured. For complete information on how to install and configure NatQuery, please refer to the *NatQuery Installation and Operations* manual, in particular the **Initial NatQuery Configuration** section.

At the point in time where NatQuery has been properly configured to handle data extraction against a particular ADABAS file of interest, all information will be present to allow the generation of the required NATCDC / NATCDCSP processing for a particular file or set of files.

Specifically, NatQuery will require:

- **Data Definition Modules (DDMs)**
  DDMs describe the layout of ADABAS files in terms that NATURAL understands. These DDMs must precisely match the layout of the corresponding File Definition Tables (FDTs) for these same source files (NatQuery / NatCDC can assist in this task).

- **Occurrence Information**
  For any DDM that references recurring fields such as Multi-Valued Fields (MUs) or Periodic-Group Fields (PEs), information must be entered or captured that describes the maximum number of occurrences of each of these fields.

- **Descriptor Statistic Information**
  For each DDM that contains one or more Descriptors, Super-Descriptors or Sub-Descriptors, information must be entered or captured that describes the dynamics of these keys / access paths. While precise information is needed and required for selective extraction that is sensitive to performance, for PLOG processing this information is non-essential and can be handled by NatQuery-assigned defaults.

- **Sign Byte Information**
  For any DDM that contains numeric-fields that can contain negative values, these fields will need to be identified so that subsequent processing can provide a sign-byte to the final output.

- **Verified Environment**
  A NatQuery Verify Environment Configuration process is executed that reports the given file is **Verified**, or not, for use.

NATCDC / NATCDCSP generation will not be usable before these items have been addressed. As it is typical that NATCDC / NATCDCSP process generation (PLOG processing) will follow the verification of a similar set of points for NatQuery, these points might only need to be reviewed at this time. In the situation where NatQuery has not been used for extraction, or perhaps in the situation where only NATCDC is purchased, the work needed to configure NatQuery to provide the information necessary for NATCDC / NATCDCSP generation can usually be accomplished in a short amount of time. A stand-alone installation will include a restricted version of NatQuery to serve as a “front-end” for NatCDC processing.

Utilization of DDMs

Of ***critical*** interest to PLOG processing is the fact that NatWorks products utilize DDMs as the mechanism that defines any given ADABAS file's content and layout, due to the simple fact that DDMs contain "user-friendly" long field names. While ADABAS file FDTs precisely reflect the layout of a given ADABAS source file, FDTs reference individual fields only by their internal two-character name, and these two-character names do not typically allow for individual fields to be quickly recognized .

By utilizing the long field names available in DDMs, both NATQUERY and NATCDC users have the benefit of more easily identifying required fields. The use of DDMs however introduces a "risk" when DDMs are used for PLOG processing, due to the fact that DDMs are not required to precisely reflect the exact field layout shown in a FDT. In point of fact: DDMs can be built that have a different field order than what is physically shown in the FDT, DDMs are not required to reference all fields that exist in an FDT, and DDMs can even have different field definitions for individual fields.

For extraction purposes, the fact that DDMs do not always precisely match FDTs is no issue since NATURAL transparently handles this. For PLOG processing however, understanding the precise order of fields is absolutely critical in order to correctly map the fields appearing in a PLOG variable-length record into fixed-length format. For NATCDC / NATCDCSP processing then, it is a requirement that DDMs ***MUST*** precisely match the layout of the file's FDT.

While having DDMs that exactly match FDTs can be completely handled manually, NatQuery has the automated ability to insure that this is the case. Specifically, NatQuery has the ability to automatically download a DDM for a given file, has the further automated ability to download the FDT for a given file, and can then automatically "merge" the DDM information into the FDT layout to create a "new" DDM. This "new" DDM, which resides only within the workstation environment (I.E. – nothing needs to change in NATURAL in the server environment), will exactly match the FDT field layout and definitions while retaining the benefit of a DDM's "user-friendly" names.

In order to handle the generation of NATCDC or NATCDCSP processing there are several configuration steps that must take place in NatQuery, and an overview of this process is described in the subsequent section entitled Overview of Configuring DDMs / FDTs.

With DDMs configured and Verified for use by NatQuery, the Administrator can then proceed with the steps to generate the required PLOG processing.

NATCDC versus NATCDCSP Generation

As described previously, the NATCDC module is used when there is a need to process PLOG transactions against a single ADABAS file in a single pass of a PLOG file. Alternatively, NATCDCSP is used when there is a need to process PLOG transactions against multiple ADABAS files in a single pass of a PLOG file.

With either approach, the specifications for how the PLOG data should be processed for a given file must be created so that generation can react to that information. Rules for creating and or modifying these specifications are outlined in the section entitled Generating NATCDC Objects for a Single File.

During the process of Generating NATCDC Objects for a Single File, the administrator's **Processing Option** is required to be, one of two choices: **Stand Alone Process** or **Component of Multi-File Process.**

If the **Processing Option** is set to **Stand Alone Process**, then the result of Generating NATCDC Objects for a Single File will be the generation of all objects necessary to process the PLOG transactions for a single file in a single pass of a PLOG file (a NATCDC process).

If the **Processing Option** is set to **Component of Multi-File Process**, then the result of **Generating NATCDC Objects for a Single File** will be the generation of all objects necessary to process the PLOG transactions for the file being defined (with the exception of required JCL / Script files), and the given file will become associated to a user-specified "**Process Name**". This **Process Name** then allows for the subsequent generation of a PLOG process that can handle the PLOG transactions against some or all of the files which share the same **Process Name** (a NATCDCSP process). The generation of this single process is described in the section entitled Generating NATCDCSP Objects for Multiple Files.

Overview of Configuring DDMs / FDTs

In order for NATCDC / NATCDCSP to "see" DDMs / FDTs for a given ADABAS file and then allow these files to be processed, the DDMs of interest must exist within the NatQuery environment and be Verified for use.

The process of making a given DDM available to NatQuery, as well as the process of insuring that the given DDM is Verified, is fully outlined in the *NatQuery Installation and Operations* manual; specifically the section entitled **Initial NatQuery Configuration**.

The following is a brief overview of the steps required to allow NATCDC / NATCDCSP to "see" and process a given DDM / FDT:

1. **Download / Import DDMs of Interest into NatQuery**
   Using the **Download DDM** function of NatQuery, the DDMs that represent the files that will be processed by NATCDC / NATCDCSP are captured and imported into NatQuery.

   Assuming that NatQuery has been properly configured and is able to interact with the remote NATURAL server through FTP or Network Copy operations, then DDMs are captured by first clicking on **Administer** drop-down menu, then clicking on **Environment Configuration**, then clicking on **DDMs / FDTs**, and then clicking on **Download DDM**. This action will result in the presentation of the **Download DDM** window that will allow for the download of the desired DDMs of interest.

   Using functions of NatQuery, these DDMs can then be automatically imported, or can alternatively be manually imported.

2. **Download / Import FDTs into NatQuery**
   Unless the Administrator is ***ABSOLUTELY SURE*** that the DDMs that have been downloaded ***EXACTLY MATCH*** the corresponding FDTs for the same file, the Administrator should now utilize the **Download FDT** function of NatQuery to download the FDTs that relate to the DDMs just downloaded. This processing allows NatQuery to merge the previously downloaded DDM information into the layout seen in the FDT, resulting in a "new" DDM that exactly matches the FDT's layout (but with "User-Friendly" long field names).

   Assuming that NatQuery has been properly configured and is able to interact with the remote NATURAL server through FTP or Network Copy operations, FDTs are captured into NatQuery by first clicking on the **Administer** drop-down menu, then clicking on **Environment Configuration**, then clicking on **DDMs / FDTs**, and then clicking on **Download FDT**. This action will result in the presentation of the **Download FDT** window that will allow for the download of the desired FDTs of interest.

Once an FDT is downloaded, and assuming that there exists a DDM in the NatQuery Environment path which has the same File Number (FNR), NatQuery will then perform the logic necessary to merge the existing DDM into the layout of the just downloaded FDT.

When handling FDTs with NatQuery, it should be noted that all FDTs imported by NatQuery will be internally converted into a DDM format.

3. **Provide Occurrence Information for DDM(s)**
   Once DDMs are present in NatQuery's Environment Path, NatQuery will require that these DDMs have **Occurrence Information** provided if the DDM contains recurring fields (MUs or PEs). This information becomes critically import when handling ADABAS files through NATCDC / NATCDCSP, as the primary function of these core programs is to transform variable-length records into fixed-length records. This transformation process requires that the desired number of occurrences for each recurring field be defined to NatQuery.

   If the administrator is uncertain of the number of occurrences used for any MU or PE in a given file, the NatQuery **Data Discovery / Analysis** function called **Repeating Field Analysis** may be used to either sample a given file or review the full file to determine this information. This function is available to an Administrator by first clicking on the **Administer** drop-down menu, then clicking on **Environment Configuration**, and then clicking on **Data Discovery / Analysis**, and then clicking on **Repeating Field Analysis**. Subsequent to using this function, NatQuery will generate the Natural program to gather the requested MU / PE usage information.

   To provide **Occurrence information** for a DDM, the **Administer Occurrence Information** function is utilized. This function is available by first clicking on the **Administer** drop-down menu, then clicking on **Environment Configuration**, and then clicking on **Occurrence Information**. This action will invoke the **Administer Occurrence Information** window that will allow for the creation of **Occurrence Information** for any known file (DDM). The **Occurrence Information** required can either be directly imported from DDMs (if DDMs are generated with **Occurrence Information** in remarks), manually entered, or as the results of an execution of **Repeating Field Analysis** can be imported.

4. **Provide Descriptor Statistics for the DDM(s)**
   Descriptor Statistics are typically used by NatQuery in its capacity as an extraction engine such that NatQuery has a means to determine which Descriptor / Super-Descriptor would yield the optimal access path to resolve a given data extraction request.

   When using NatQuery for Data Warehousing (DWH), or for using NatQuery with NATCDC, these **Descriptor Statistics** become of nominal importance given the fact that

- in most cases – data extracts will be done with no regard for existing access paths. This is because with most DWH extractions for a given file, the entire file is likely to be read and extracted (and in that case a Read Physical will vastly outperform a Read Logical or Find), as opposed to "selective" extraction which extracts subsets of records from a given file (in which case Read Logical or Find may be more suitable).

While **Descriptor Statistics** are of nominal use for full-file DWH extraction or for NATCDC processing, **Descriptor Statistics** must still exist for any file that will be used by NATCDC (due to the requirement that a given DDM must be Verified for use by a NatQuery Verify Environment Configuration process, a process that checks for the existence of **Descriptor Statistics** for each file / DDM).

In the case where a given DDM / File is desired to be used for NATCDC / NATCDCSP or DWH applications only (I.E. NatQuery will not be used for End-User data extraction), it is suggested that the **Descriptor Statistics** for the given file be provided simply as all zeros. Using this approach, files can be quickly verified without the requirement to capture or enter accurate **Descriptor Statistics**, as would be expected when NatQuery is to be used for customized End-User extraction requests.

To provide **Descriptor Statistics** for a given DDM, the **Administer Descriptor Statistics** function is used. This function is available by first clicking on the **Administer** drop-down menu, then clicking on **Environment Configuration**, and then clicking on the **Descriptor Statistics**. This action will invoke the **Administer Descriptor Statistics** function that will allow for the handling and capture of required **Descriptor Statistics**.

If necessary, the Administrator has the option of using NatQuery to generate a NATURAL program that will accurately determine what the **Descriptor Statistics** for a given file should be, with the results then being importable into the NatQuery **Descriptor Statistics** function.

5. **Review Sign Byte Information for each DDM**
When the NatQuery and NATCDC / NATCDCSP processes handle fields that are numeric in nature, NatQuery assumes that all such fields are positive and that no sign position is therefore required in the final output for these fields (thus potentially saving a great deal of space in output).

   If there are numeric-type fields that may be negative within a given DDM, then NatQuery needs to be informed such that when these fields are output by either NatQuery or NATCDC, these fields can be output with a leading sign position.

   To define **Sign Byte** information to NatQuery, the **Administer Sign Byte Information** function is used. This function is available by first clicking on the **Administer** drop-down menu, then clicking on **Environment Configuration**, and then clicking on **Sign**

Byte Information. This action will invoke the **Administer Sign Byte Information** window that will allow for the designation of which fields will require a sign byte in the final NatQuery or NATCDC / NATCDCSP output.

6. **Verify the Environment Configuration**
   After the above steps are completed for each DDM of interest for PLOG processing, the final step prior to using NATCDC / NATCDCSP generation functions is to Verify the Environment Configuration for correctness.

   This is accomplished by using the **Verify Environment Configuration** function of NatQuery. This function is available by first clicking on the **Administer** drop-down menu, then clicking on **Environment Configuration**, and then clicking on **Verify Environment Configuration**. This action will invoke a Verification process that will result in the creation of a report that details the overall status of the current Environment Configuration.

   Once the report is created, the Administrator should review the report to insure that no outstanding issues concerning any of the files of interest to NATCDC / NATCDCSP processing are found. If issues are found during the verification process, these issues will be detailed in the report, and the Administrator should then take appropriate action(s) to correct any deficiencies.

With the above-summarized steps accomplished, the specific DDMs of interest will now be available for use with NATCDC / NATCDCSP.

Generating NATCDC Objects for Single File

Once DDMs have been verified through the **Verify Environment Configuration** function of NatQuery, these DDMs can then be used to generate required NATCDC / NATCDCSP objects (such as parameter files, programs, JCL / Script, and any other required objects).

The entry and modification of the processing specifications occurs through the use of the **Generate NATCDC Objects for Single File** function. As a pre-requisite to using the **Generate NATCDC Objects for Single File**, the Production NATCDC Process JCL template must exist; even if the intent of the administrator is to generate a Single-Pass process that handles multiple files. For information on handling this JCL / Script, please refer to the section entitled Configuring JCL / Script.

To generate the PLOG processing for a specific ADABAS file, perform the following:

1. **Start NatQuery**
   If NatQuery is already started, then insure that NatQuery is on an "open" desktop (I.E. no queries open and no Administrative windows are displayed.

2. **Invoke the Generate NATCDC Objects for Single File function**
   The **Generate NATCDC Object for Single File** function is accessed by clicking on the **Administer** drop-down menu, then clicking on **NATCDC / NATCDCSP**, and then clicking on **Generate NATCDC Objects for Single File**.

3. **Generate NATCDC Objects for Single File - Select File of Interest**
   In the upper left-hand corner of the **Generate NATCDC Objects for Single File** window there is a selection-box labeled "**Generate NatCDC Objects for**", with this selection box allowing for the selection of a DDM that has been verified for use.

   To assist in the creation of consistent NATCDC specifications for files of interest, the **Generate NATCDC Objects for Single File** function supports the concept of a **Default Profile**. By creating a **Default Profile**, this profile will be automatically applied to NATCDC specifications being created against files for the first time, thereby allowing for the rapid creation of a NATCDC specification.

   To create or modify a **Default Profile** for NATCDC processing, the administrator will select the "**<Default-NatCDC-File-Profile>**" from the **Generate NatCDC Objects for** selection control. When this item is selected, the administrator can then proceed through the Tabs and related graphical controls described in the following steps - except for those controls which are not "generic" such as File Number. When all information for the **Default Profile** has been created or modified as appropriate, the administrator can then repeat this step and the following steps for the creation of new specifications against required files, with each file's specifications initially defaulting to the options entered into the **Default Profile**.

To proceed to the following steps, the administrator must make a selection from the **Generate NATCDC Objects for** selection control, be it the Default Profile or an actual file name.

Immediately after an entry is selected from the **Generate NATCDC Objects for** selection control, the user will be presented with a message box that indicates whether or not the selection has been handled previously with this function or not. This is strictly informational, but for those instances where a new specification is being created and a Default Profile has been created – the administrator will be informed that the Default Profile is being applied.

If the appropriate file of interest is not found in the selection control, then this indicates that either the appropriate DDM is not present in the current NatQuery Configuration path, or this DDM exists but is not verified. In either case: The Administrator should execute the **Verify Environment Configuration** function to ascertain the cause of the problem. If the file is not shown in the Verification Report, then the DDM (and possibly FDT) should be imported into NatQuery as described in Overview of Configuring DDMs / FDTs. If the DDM is shown in the Verification Report but not in the selection control, then the DDM is not properly configured. The Verification Report produced by the Verify Environment Configuration function will detail what is required to get the file of interest Verified For Use, after the corrections are applied and re-verified it will become available in the selection box of the **Generate NATCDC Objects for** selection control.

The **Generate NATCDC Objects for Single File** categorizes the various pieces of information that pertain to NATCDC's processing of a file through the use of tabs. By default, the **File Information** tab will be displayed first.

4. **Generate NATCDC Objects for Single File – File Information Tab**
On the **File Information** tab, review the following:

4.1. **File Information Tab – Database / File Number Frame - DBID**
Enter the Database ID corresponding to the appropriate ADABAS database that contains the selected file of interest into the **DBID** text field. The entered DBID must be five characters and be numeric, so this may require that the DBID value should be padded with leading zeros as appropriate.

4.2. **File Information Tab – Database / File Number Frame - File Number**
Insure that the **File Number** of the selected file matches the value supplied in the **File Number** field with leading zeros as appropriate (this field should now be populated by default).

NOTE: If handling a **Default Profile**, this control will not be available.

4.3. **File Information Tab - Expanded File Information Frame**
If the file of interest ***IS NOT*** an expanded file (most files will not be expanded files) then continue with the next step by clicking on the **Field Information** tab.

If however the file ***IS*** an expanded file, then the **File is an Expanded File** checkbox should be selected, and the appropriate information regarding the "chained" files should be entered into the **Component Information** grid.

This task involves entering the **File Number** and **Min** (minimum) **ISN** for each chained file from the base file, until all information on all chained files is entered.

Once this is done, then the user will click on the **Field Information** tab.

NOTE: If handling a **Default Profile**, this control will not be available.

5. **Generate NATCDC Objects for Single File – Field Information Tab**
Through the **Field Information** tab, the user can indicate two things: Whether or not the selection of fields that will be processed from a PLOG should be put into sync with a previously generated NatQuery Extract (**Synchronize Output Fields to Query**), as well as indicating which fields should be omitted / dropped from the final output file (if any).

If NatQuery has been previously used to create an extract against the selected file and it is desired that a PLOG process be generated that handles exactly the same fields, then the name of this query should be selected in the selection-box labeled **Synchronize Output Fields to Query**. If no such extract exists, then this control can be ignored.

By default, all "real" fields will be selected as being included in the final output, with the exception of "C*" fields (fields that are prefixed with a "C*").

The processing of "C*" fields relate to Multi-Values fields (MUs) and Periodic-Group (PEs), with the purpose of these "C*" fields being to contain the actual number of occurrences that existed on the source ADABAS record. Under most circumstances, C* fields are not needed, and are therefore omitted by default.

When the file's fields have been reviewed for inclusion / omission, the user can click the **Output Options** Tab.

NOTE: If handling a **Default Profile**, this Tab will not be available.

6. **Generate NATCDC Objects for Single File – Output Options Tab**
The **Output Options** tab controls several aspects of how a PLOG process will handle the output of data. Each of the available **Output Options** is discussed below.

6.1. Output Options Tab - Format Conversion Frame

The **Format Conversions** option is used to generate the PLOG processing that either automatically performs **Format Conversions** on source fields that may require **Format Conversions**, or to bypass **Format Conversions** completely and leave fields in their native format. In most cases, it is desirable and recommended to perform **Format Conversions**, and therefore the **Perform Conversions** option is selected by default.

Whenever data is being moved from a mainframe (EBCDIC) environment to a UNIX, Linux, or Windows environment (ASCII); it is required that the **Perform Conversions** option be selected.

6.2. Output Options Tab - Data Retrieval Options Frame

If it is desired that leading zeros in numeric fields be suppressed, then the option Suppress leading zeros in numeric output should be selected (Checked).

In most cases, it is NOT desirable to suppress leading zeros in numeric output, and this option would typically be left un-checked.

6.3. Output Options Tab - Logical Record Handling Frame

The **Logical Record Handling** option controls how individual transactions will be handled. By default, the **Logical Last Image (Delta)** option is selected, as it is anticipated that this option will suit most PLOG processing needs.

The choices for Logical Record handling are:

6.3.1. Logical Last Image (Delta)

The **Logical Last Image (Delta)** option instructs NatQuery to generate a PLOG process so that only a single record will be presented in the final output for each logical grouping of transactional records related to the same record / ISN, regardless of how many physical transactions may have occurred against a specific record / ISN in a PLOG.

This **Logical Last Image (Delta)** will contain the last image of the record as it stood at the end of the PLOG time span, and will contain a flag field in the header portion of the output data that reflects the summary action that occurred (**S**tore, **U**pdate or **D**elete).

Example #1

A record was Stored and then Updated 2 times across a given PLOG time span. The **Logical Last Image (Delta)** option will provide a single record whose image reflects the image of the record from the last **U**pdate, with the flag indicating a **S**tore.

Example #2
A previously existing record was Updated 3 times. The **Logical Last Image (Delta)** option will provide a single record image that reflects the record image that resulted from the last **U**pdate, and the flag of this record will indicate an **U**pdate.

Example #3
A previously existing record was Deleted in a PLOG time span, with or without intervening Updates. The **Logical Last Image (Delta)** option will provide a single record image that reflects the image of the Deleted record with a flag indicating a **D**elete.

Example #4
A unique record is Stored and then Deleted, with or without intervening Updates, within the same PLOG time span. The **Logical Last Image (Delta)** will contain no record in the final output (this is because it is as if the record never existed).

NOTE:
The **Logical Last Image (Delta)** processing IS sensitive to ISN re-usage that may be set as an option for an ADABAS file. This means that even though ISN is used by NATCDC / NATCDCSP processing to identify a unique record – this processing is completely aware that the same ISN can occur multiple times in a given PLOG and represent completely different records.

6.3.2. First and Last Images

The **First and Last Images** option of **Logical Record Handling** instructs NatQuery to generate PLOG processing that typically produce two records for a given unique record / ISN: The first image found for the record in the PLOG and the last image found, regardless of how many intervening transactions may have occurred. The first record output will reflect the record image at the beginning of the PLOG time span and the second record output will reflect the record image at the end of the PLOG time span. In this case, the second record will indicate the status of the last transaction (Store, Update, or Delete).

6.3.3. All Images

The "**All Images**" option of **Logical Record Handling** instructs NatQuery to generate PLOG processing that outputs all records read, typically sorted by ISN and Date / Time Sequence, a format ideally suited for auditing purposes.

In this case, each record will be flagged with a "B" (to represent a PLOG "Before" image or an "A" to represent a PLOG "After" image".

6.4. Output Options Tab - Data Warehouse Integration Frame

The options shown in the **Data Warehouse Integration** frame control generational

options that can come into play when integrating ADABAS PLOG data into other target platforms & databases.

When landing PLOG data into Extraction, Transformation, and Loading (ETL) tools, NATCDC / NATCDCSP has the ability to generate “interface files”, or external files that describe format and layout of the NATCDC / NATCDCSP provided data in terms that ETL tools can understand. This allows an organization to seamlessly and automatically load their ADABAS PLOG data into most existing or new ETL toolsets.

Alternatively, both NATCDC and NATCDCSP processing can be generated that directly integrate ADABAS PLOG data into Oracle or SQL Server targets.

Each of these targets are described below:

6.4.1. DSX Files (ETL Tool)

If the primary purpose of running NATCDC / NATCDCSP processing is to provide ADABAS record changes to a third-party product such as IBM’s WebSphere Integration Suite (formerly Ascential Software’s DataStage product suite) then the **Data Warehouse Integration** option should be set to **Enabled** (checked) with **DSX** being the selected **Method** of integration.

In response to selecting DSX, when generation occurs NatQuery will generate a file in the proprietary DSX format that will describe the NATCDC / NATCDCSP final record layout to DataStage / WebSphere in terms it immediately understands.

6.4.2. CFD (ETL Tool)

If the primary purpose of running NATCDC / NATCDCSP processing is to provide ADABAS record changes to a third-party product that can import or otherwise understand a COBOL File Definition (CFD) to describe the final layout of the NATCDC / NATCDCSP data, then the **Data Warehouse Integration** option should be set to **Enabled** (checked) with **CFD** being the selected **Method** of integration.

In response to selecting CFD, when generation occurs NatQuery will generate a CFD file that describes the NATCDC / NATCDCSP final record layout.

6.4.3. Oracle (Direct Integration)

If the intent of running a NATCDC / NATCDCSP process is to have a generation that will handle the delivery of PLOG data for a specific file or files AND have the ability to automatically integrate that data into **Oracle**, then the **Data Warehouse Option** should be set to **Enabled** (checked) with **Oracle** being the selected Method of integration.

In response to selecting Oracle, when generation occurs NatQuery will generate the Script or batch file, DML, SQL and related objects that combined will provide a complete solution for handling ADABAS PLOG data into Oracle.

6.4.4. SQL Server (Direct Integration)

If the intent of running a NATCDC / NATCDCSP process is to have generation that will handle the delivery of PLOG data for the specific file AND have the ability to automatically integrate that data into **SQL Server**, then the **Data Warehouse Option** should be set to **Enabled** (checked) with **SQL Server** being the selected **Method** of integration.

In response to selecting SQL, when generation occurs NatQuery will generate the Script or batch file, DML, SQL and related objects that combined will provide a complete solution for handling ADABAS PLOG data into **SQL Server**.

6.5. Output Options Tab - ADASEL Output Option (Mainframes Only) Frame

If NATCDC / NATCDCSP is used against ADABAS that executes on a UNIX, Linux or Windows server, then the user may now proceed to the **General Options** tab by clicking on it.

This option is for backwards compatibility with versions of NATCDC prior to 3.2.0. Beginning with 3.2.0, NATCDC was modified so that it expected to see output from the ADABAS ADASEL utility that was created using the EXTENDED parameter. Versions prior to 3.2.0 expected to see output from the ADASEL utility created with the older LOGINFO parameter.

It is strongly *suggested* that all new installations of NATCDC / NATCDCSP use the EXTENDED option, and all previous older versions of NATCDC be upgraded to 3.2.0 or higher and begin using the EXTENDED option.

This option is only available when NatQuery is configured to have a Server Type of MVS or VSE – and is therefore only available for use against mainframe installations of ADABAS.

Support for UNIX, Linux and Windows does not use the ADASEL utility, it uses ADAPLP instead, so this option is not available against these servers.

When the appropriate options have been selected on the **Output Options** tab, the user can click on the **General Options** tab.

7. Generate NATCDC Objects – General Options Tab

The **General Options** tab controls the use of general options relating to NATCDC /

NATCDCSP processing. These options are described below.

7.1. **General Options Tab - Time Differential Handling Frame**

The **Time Differential Handling** controls are used to control situations where a PLOG has been generated in one time zone, but now requires processing in another time zone. This option then essentially controls how Dates will be presented in the final NATCDC output relative to the time zone that the processing is occurring in. In almost all cases, this option should be left at its default of **Auto**.

7.2. **General Options Tab - DDM Generation Frame**

The **DDM Generation** option instructs NatQuery to generate a DDM file that describes the layout of the final output file in a DDM format. With this DDM generated, it is then possible to Import the DDM into NatQuery where it is then possible to use NatQuery to generate a Natural program to process / query this file.

7.3. **General Options Tab - Totals File Generation**

The Totals File Generation option controls whether or not a NATCDC process will create an ancillary file that provides record processing information.

If selected, a NATCDC process will produce a one-line file that details what the processing accomplished. The format of this line will change based on the setting of **Logical Record Handling** as described above.

If **Logical Record Handling** is set to **Logical Last Image (Delta)** or is set to **First and Last Images**, then the line output will have the form of:

```
99999999999 Total, 99999999999 Stores, 99999999999 Updates, 99999999999 Deletes
```

In the above output example: “Total” refers to the total number of records that were read from the PLOG utility output, “Stores” refers to the number of Store transactions encountered “Updates” refers to the number of Update transactions encountered, and “Deletes” refers to the number of Deletes encountered.

If the setting of **Logical Record Handling** is set to **All Images**, then the line output will have the form of:

```
99999999999 Total, 99999999999 BI Images, 99999999999 AI Images
```

In the above output example: “Total” refers to the total number of records that were read from the PLOG utility output, “BI Images” refers to the total number of Before Image (BI) record images encountered, and “AI Images” refers to the number of After Images (AI) record images encountered.

7.4. **General Options Tab - Single Pass Processing**

The Single Pass Processing is a critical option that instructs the generation process as to whether a NATCDC process is to be generated (a single process that handles only the ADABAS PLOG transactions from the specified file) or whether a NATCDCSP process will be generated (a single process that handles the ADABAS PLOG transactions for the specified file and possibly other files as well).

Stand Alone Process

If the intent is to generate a single JCL / Script process that will only handle the specified ADABAS file's PLOG transactions when executed, then the **Stand Alone Process** option should be selected. When generation begins, this option will cause all objects required for immediate execution to be generated, including the required JCL / Script, supporting parameter file, required post-NATCDC NATURAL program and any other objects possibly needed.

Component of Multi-File Process

If the intent is to generate the objects necessary to include PLOG transactions from the currently specified file into a single process that handles the PLOG transactions of other files as well, then the **Component of Multi-File Process** option should be selected. When this option is selected, then the administrator will be given the option of either creating a **Process Name** that the specified file will be part of, or selecting an already defined **Process Name**. The created or selected **Process Name** then links the PLOG processing of the specified file to the named process – allowing a separate function called Generate NATCDCSP Objects for Multiple Files (the use of which is described in a following chapter)**.**

When **Component of Multi-File Process** is selected, then when generation begins this option will cause the generation of only the parameter file and file-specific NATURAL post-NATCDCSP processing program, with the generation of JCL / Script, parameter files and supporting files deferred until the Generate NATCDCSP Objects for Multiple Files function is performed.

7.5. **General Options Tab - ADABAS Version (UNIX and Linux only) Frame**

To handle the minor differences in ADAPLP output in UNIX environments between ADABAS version 3.1 and 3.2 or higher, this option allows the version of ADABAS to be supplied. The resulting generation will therefore be aware of how to handle the generation across these versions.

NOTE: This option is not available when the **Server Type** is MVS or VSE.

8. **Generate NATCDC Objects – File Names Tab**

The **File Names** Tab provides NatQuery with information pertaining to the **File Names Referenced in JCL / Script**, the **File Names used in FTP / Copy**, and the **Name of the**

post-NATCDC / NATCDCSP Program (or sub-program) that will be generated.

In using this Tab, it should be noted that the **File Names Referenced in JCL / Script** and the **File Names used in FTP / Copy** may not be available if **Component of Multi-File Process** has been selected as a **Single Pass Processing** option. This is because these fields are only relevant when generating a NATCDC **Stand Alone Process**, with the field names required for a NATCDCSP Multi-File Process being supplied when the appropriate **Process Name** is generated using the Generate NATCDCSP Objects for Multiple Files function described later in this manual.

8.1. **File Names Tab - File Names Referenced in JCL / Script Frame**

The **File Names Referenced in JCL / Script** fields are the values of the file names that will be used in a Stand Alone Process for substitution into the Production NATCDC Process JCL / Script template. No file names will be available if **Single Pass Processing** is set to **Component of Multi-File Process**. For important information concerning the use of these file names, please see the **NOTE**(s) later in this step.

Essentially, the file names available on this frame will be directly substituted into corresponding Dynamic Variables that reside in the Production NATCDC Process JCL / Script template, thereby allowing that generic template to become specific to the file processing required.

The file names displayed in the **File Names Referenced in JCL / Script** frame will come from the **<Default-NATCDC-File-Profile>** (if this profile exists AND this is the first time the specified file is handled by this function), from the last values entered into this form for the specified file (if the specified file has been successfully processed through the **Generate NATCDC Objects for Single File** function previously), or they can come from internally supplied defaults (that are sensitive to target environment).

Specific information on each of the fields in the **File Names Referenced in JCL / Script** is described following these important notes**.**

NOTE #1 – Initial Naming:
In most instances, and particularly with the first initial use(s) by the Administrator of the **Generate NATCDC Objects for Single File** function (and unless the administrator is creating or modifying a **<Default-NATCDC-File-Profile>**), it is advisable that the administrator not become too consumed with the naming being used. Once the final JCL / Script is generated and executed, the administrator will better understand how these respective files are used in the final JCL / Script, and will therefore have a more solid understanding of how file naming operates. If the administrator at any time finds that these file names require changing either immediately or after execution, it is STRONGLY advised that the **<Default-NATCDC-File-Profile>** be created or is otherwise modified so that acceptable defaults are always initially provided. This will

allow for consistency in file naming to be achieved, and will greatly speed the creation of specifications for other needed PLOG file transactions.

NOTE #2 – Handling the <Default-NATCDC-File-Profile>:
When handling the **<Default-NATCDC-File-Profile>**, an additional control will be displayed at the bottom of the **File Names Referenced in JCL / Script** window, with this control being named **Dynamic Substitution Values**. These values are:

&&DBID
The &&DBID tag, when used in the naming of a file reference in the **<Default-NATCDC-File-Profile>**, will be substituted for the value of the DBID (Database ID) of a given file when the specifications for a given file are initially generated.

&&FNR
The &&FNR tag, when used in the naming of a file reference in the **<Default-NATCDC-File-Profile>**, will be substituted for the value of the FNR (File Number) of a given file when the specifications for a given file are initially generated.

&&PROGRAM-NAME
The &&PROGRAM-NAME tag, when used in the naming of a file reference in the **<Default-NATCDC-File-Profile>**, will be substituted for the value of the **Post-NATCDC / NATCDCSP Program Name** given to the generated NATURAL processing program when the specifications for a given file are initially generated.

&&USER-ID
The &&USER-ID tag, when used in the naming of a file reference in the **<Default-NATCDC-File-Profile>**, will be substituted for the value of the administrator's User ID, as specified in the NatQuery Configuration function, when the specifications for a given file are initially generated.

Descriptions of File Names
A description of how each of the file names that appear in the **File Names Referenced in JCL / Script** frame are as follows:

8.1.1. PLOG Data (Input)

This file name references the PLOG Data file that is output from the initial ADABAS PLOG utility processing. For mainframes, this is the name of the file which contains the output of ADASEL (but could be the output of ADACDC), and for Open System environments this would be the output of ADAPLP and NATPLP.

In JCL / Script templates, this field equates to the Dynamic Substitution Variable **&&ADASEL-OUTPUT**.

NOTE #1:
At the time of initial generation for a specific file, this file may not physically exist yet, and would not exist until the appropriate PLOG utility program(s) is run.

NOTE #2:
The use of the name **&&ADASEL-OUTPUT** will be the name used for dynamic substitution even though the PLOG was created on an Open System and is then processed through the ADAPLP utility.

8.1.2. NATCDC Parameters (Input)

This file name references the Parameter File that will be generated for the specified file, with this Parameter File instructing a NATCDC / NATCDCSP process on how to handle the mapping of that file from a PLOG record.

In JCL / Script templates, this field equates to the Dynamic Substitution Variable &&CDC-PARMS.

8.1.3. Intermediate File #1 (Temp)

This file name references a temporary file that is only used to contain transactional information between processing steps.

In JCL / Script templates, this field equates to the dynamic substitution variable &&CDC-I-DSNAME1.

8.1.4. Intermediate File #2 (Temp)

This file name references a temporary file that is only used to contain transactional information between processing steps in the JCL / Script.

In JCL / Script templates, this field equates to the dynamic substitution variable &&CDC-I-DSNAME2.

8.1.5. Final Data (Output)

This file name references the primary output file of a NATCDC process and is the data file that result from this process.

In JCL / Script templates, this field equates to the dynamic substitution variable &&CDC-F-DSNAME.

8.1.6. Optional Totals File (Output)

This file name references an optional output file of a NATCDC process and is the Totals file that can result from this process. See step 7.3 above for further information on the content of this file.

In JCL / Script templates, this field equates to the dynamic substitution variable &&CDC-F-DSNAME.

8.2. **File Names Tab - File Names to be used in FTP/Copy Frame**

The **File Names to be used in FTP/Copy** fields are the values which will be applied to the file names of the generated JCL / Script and the generated Parameter File when these files are placed onto the server by automated FTP or File Copy processing.

When creating PLOG processing against a mainframe environment, then in almost all cases the file names entered into these two fields will be identical to the names of these same files in 8.1 above, with these fields allowing the JCL file and the Parameter file to be moved under different references. This would then handle the situation where FTP against a server establishes a working directory by default, and the administrator wishes to have the files moved into a different directory.

NOTE (Script Generation):
When handling the generation of Script against Open Systems, NatWorks follows the convention suggested by Software AG for Batch execution, meaning that in addition to a base Script being generated, additional files will be generated that support that Script: Specifically a COMBJIN and a CMSYNIN file will be generated. The base script will refer to these files, and these files will be moved together with the base Script onto the server platform if the software is allowed to. Therefore the field labeled **NATCDC Script File Prefix** will be used as the file name prefix for the Script as well as the CMOBJIN and CMSYNIN files, with “sh” or “bat” file extension for the Script, and “txt” extensions for the CMOBJIN and CMSYNIN files.

8.3. **File Names Tab – Post-NATCDC / NATCDCSP Program Name Field**

Subsequent to the PLOG processing that will occur by either NATCDC or NATCDCSP, the data needs to be processed by a generated NATURAL program so that all remaining processing can be completed. The **Post-NATCDC / NATCDCSP Program Name** field allows the administrator to provide the name of this program.

NatQuery will provide a default name for the **Post-NATCDC / NATCDCSP Program Name**, with this name beginning with a hard-coded value of “CDC” followed by a NatQuery assigned unique number.

The administrator should review the provided **Post-NATCDC / NATCDCSP Program Name**, and make any changes to it as required.

NOTE #1:
It should be remembered that when NatQuery assigns the program name, it does so by only considering the names in use by NatQuery for PLOG processing that exist in the

NatQuery environment – it currently has no ability to check the use of program names in the remote NATURAL environment. Therefore, it is *suggested* the program name being used should be scrutinized by the administrator for any possible naming issues.

NOTE #2:
When the **Generate NATCDC Objects for Single File** is used to generate **Single Pass Process** which is a **Stand Alone Process**, the **Post-NATCDC / NATCDCSP Program Name** will be the name of the program that is generated as a text file on the NatQuery workstation, and will become the name of this program in NATURAL on the server if NatQuery processing is allowed to introduce it. When the **Generate NATCDC Objects for Single File** is used to generate a **Single Pass Process** which is a **Component of a Multi-File Process** however, the **Post-NATCDC / NATCDCSP Program Name** will be the name of the sub-program that is generated as a text file on the NatQuery workstation, and will become the name of this sub-program in NATURAL on the server if NatQuery processing is allowed to introduce it.

9. **Generate NATCDC Objects – Transactions Tab**
   The **Transactions** tab is used when PLOG processing is being generated with the Single Pass Option set to Stand Alone Process and the target system is a mainframe environment where JCL needs to reference the Disk Space that will be needed to process PLOG transactions. When a **Stand Alone Process** is being generated, then the value entered into the **Estimated Number of Transactions** will be used to calculate the allocations needed for Disk Space references in JCL.

   NOTE:
   When PLOG Processing is generated against UNIX, Linux or Windows platforms, or the Single Pass Process option is set to Component of Multi File, then the **Transactions** tab will be disabled as this information will not be needed.

10. **Generate NATCDC Objects for Single File – Wrap Up**
    When the information on all appropriate tabs has been reviewed and modified as needed, the administrator can now click the **OK** button.

    This action will invoke a number of edits against the supplied specifications, and any identified error must be corrected prior to generation or any attempted file movement.

    Subsequent prompts and message boxes will depend on the options selected, with window-specific Help being available on any given screen by clicking the **Help** button.

    At some point in the generation process, and depending upon several factors (including the method of integration being used and whether or not a **Stand Alone Process** is being generated, an administrator may wish to generate all required objects but not introduce these objects into the server environment. The generation process allows for this, and will prompt

the user for confirmation on moving generated objects to the server.

When all editing has been accomplished and all generated objects have been created, the **Generate NATCDC Objects for a Single File** will conclude its processing with the generation of a report that details what was accomplished. While being presented to the user via a scrollable text window, this report can be printed and will also be saved to disk.

Results of Generation

What is generated by the **Generate NATCDC Objects for Single File** function will depend on several factors.

Unless otherwise specified, the files that are generated into the workstation environment will be placed into the currently specified **Data Download \ Output Directory** of NatQuery (as set through the **NatQuery Configuration** function, **Environment Paths** tab).

The objects that can be generated are:

- Natural Program / Sub-Program
- Parameter File (external)
- Parameter File (internal)
- Data Warehouse Integration
  - DSX Files
  - CFD Files
  - Direct Integration to Oracle
  - Direct Integration to SQL Server
- JCL / Script
- Report
- DDM

Natural Program / Sub-Program

If generation is allowed to occur, then besides any other generation that may occur, either a NATURAL Program or Sub-Program will be generated, depending on the setting of **Single Pass Processing**. If **Single Pass Processing** is set to **Stand Alone Process** then a NATURAL program will be generated; if **Single Pass Processing** is set to **Component of Multi File Process**, then a NATURAL Sub-program will be generated.

The program or sub-program will be generated into the **Data Download \ Output Directory** and will have a file name equal to the value of the **Post-NATCDC / NATCDCSP Program Name** (as specified on the **File Names** tab) with either a "**NSP**" file extension (if a program) or a file extension of "**NSN**" (if a sub-program).

Parameter File (External)

If generation is allowed to occur, then besides any other generation that may occur, an "external" **Parameter** file will be generated that will be used as input to a NATCDC / NATCDCSP process, with the purpose of this parameter file being to described the required field mapping to the NATCDC or NATCDCSP processing modules.

The **Parameter** file will have a file name equal to the value of the **Post-NATCDC / NATCDCSP Program Name** (as specified on the **File Names** tab) with a "**CDC**" file extension.

Parameter File (Internal)

If generation is allowed to occur, then besides any other generation that may occur, an "internal" **Parameter** file will be generated that will be used to save the user's specifications for the currently selected file for subsequent re-presentation to the user when / if the file is next selected through the **Generate NATCDC Objects for Single File** function.

The **Parameter** file will have a file name equal to the value of the **Post-NATCDC / NATCDCSP Program Name** (as specified on the **File Names** tab) with a "PRM" file extension.

Data Warehouse Integration

If **Data Warehouse Integration** is **Enabled**, then besides any other generation that may occur, one or more integration files will be generated, depending upon the **Data Warehouse Integration** option selected as described below.

DSX (Data Stage Exchange File)

If the **Data Warehouse Integration** option is set to **DSX**, then a **DSX** file will be generated into a path specified by the user during generation.

By default, the DSX file name will be equal to the value of the **Post-NATCDC / NATCDCSP Program Name** (as specified on the **File Names** tab) with a "**DSX**" file extension. The default

path will take the value of the **DSX / CFD Default Path** (specified through the **NatQuery Configuration** function, **DWH Options** tab).

CFD (COBOL File Definition File)
If the **Data Warehouse Integration** option is set to **CFD**, then a **CFD** file will be generated into a path specified by the user during generation.

By default, the CFD file name will be equal to the value of the **Post-NATCDC / NATCDCSP Program Name** (as specified on the **File Names** tab) with a "**CFD**" file extension. The default path will take the value of the **DSX / CFD Default Path** (specified through the **NatQuery Configuration** function, **DWH Options** tab).

Oracle
If the **Data Warehouse Integration** option is set to **Oracle** then several files will be created. These include a SQL Script that will create the Staging Table that will initially receive the NATCDC / NATCDCSP provided data into Oracle, a SQL Script that will load the NATCDC / NATCDCSP provided data into the Staging Table and from there handle this data into the appropriate Oracle receiving table(s), a Format File that describes the layout of the NATCDC / NATCDCSP provided data in terms the Oracle Bulk Loader (SQL Loader) will understand, and a Batch (for windows) or Script (for UNIX or Linux) file that will execute the complete data handling process against SQL Server.

For further information concerning the NatWorks integration to a RDBMS, please refer to the **NatWorks RDBMS Integration Manual**.

SQL Server
If the **Data Warehouse Integration** option is set to **SQL Server** then several files will be created. These include; a SQL Script that will create a Staging Table that will initially receive the NATCDC / NATCDCSP provided data into SQL Server, a SQL Script that will load the NATCDC / NATCDCSP provided data into the Staging Table and from there handle this data into the appropriate SQL Server receiving table(s), a Format File that describes the layout of the NATCDC / NATCDCSP provided data in terms the SQL Server Bulk (BCP) Loader will understand, and a Batch or Script file that will execute the complete data handling process against SQL Server.

For further information concerning the NatWorks integration to a RDBMS, please refer to the **NatWorks RDBMS Integration Manual**.

JCL / Script
If generation is allowed to occur, then besides any other generation that may occur, a JCL or Script execution file may be generated, depending on the setting of **Single Pass Processing**. If **Single Pass Processing** is set to **Stand Alone Process** then a JCL / Script file will be generated;

if **Single Pass Processing** is set to **Component of Multi File Process**, then no JCL / Script will be generated.

Whether JCL is generated or whether Script is generated depends on the setting of **Server Type**, as specified on the **Connection Info** tab of the **Administer Server Connection Information** function. From an empty desktop, this function is accessed by first clicking on the **Administer** drop-down menu, then clicking on **Environment Configuration**, **Server Connection Configuration**, and then clicking on **Server Information**. If **Server Type** is set to **MVS** or **VSE**, then Job Control Language (JCL) will be generated; if the **Server Type** is set to **UNIX** or **NT** then Script will be generated.

The JCL / Script will be generated into the **Data Download \ Output Directory** and will have a file name equal to the value of the **Post-NATCDC / NATCDCSP Program Name** (as specified on the **File Names** tab) with a "JCL" file extension - if Server Type is MVS or VSE, "bat" - if Server Type is Windows or "sh" - if UNIX or Linux.

NOTE (Script Generation):
When generating Script for Open Systems, NatWorks products follow the recommendations of Software AG, meaning that in addition to a base Script being generated, additional supporting files are also generated, specifically CMOBJIN and CMSYNIN files. These file will be generated in parallel with the generation of the base script, the script will reference these files, and all of these files will be moved onto the server by NatQuery if the software is instructed to do so.

NOTE #2 (Script Generation):
When handling the generation of Script, the **Production NATCDC Process** template will be a single file that actually contains the entire contents of not just the Script, but also contains the entire contents of the CMOBJIN file, the CMSYNIN file and the NATURAL Program as well. If FTP or File Copy is allowed to occur, then this single file will be split out into its separate component files and these separate files will physically be created on the server platform. These "split out" component files will not be seen in the **Data Download / Output Directory** – however these can be found in the path specified as the **Environment Path**.

Report

If generation is allowed to occur, then besides any other generation that may occur, a report file will be created, with this report being the same information displayed at the completion of the **Generated NATCDC Objects for Single File** function.

The Report will be generated into the **Data Download \ Output Directory** and will have a file name equal to the value of the **Post-NATCDC / NATCDCSP Program Name** (as specified on the **File Names** tab) with a "RPT" file extension.

DDM

If generation is allowed to occur, then besides any other generation that may occur, a DDM file will be created if the **Generate DDM of final NATCDC Output** is selected. This DDM will be generated so that it exactly matches the layout of the final output data produced by NATCDC, such that this DDM can then be used by NatQuery to immediately allow Adhoc queries to be made against the completed extraction of PLOG data.

The DDM will be generated into the **Data Download \ Output Directory** and will have a file name equal to the value of the **Post-NATCDC / NATCDCSP Program Name** (as specified on the **File Names** tab) with a "NSD" file extension.

Generating NATCDCSP Objects for Multiple Files

Once the **Generate NATCDC Objects for Single File** has been successfully used at least once to generate a **Single Pass Process** type of **Component of Multi File Process**, then the **Generate NATCDCSP Objects for Multi File** function can be used to generate a PLOG process that has the ability to handle the PLOG transactions from multiple files in a single pass (a NATCDCSP process).

When used to generate a NATCDCSP process, objects needed to execute the required NATCDCSP process will be created (such as parameter files, NATURAL program, JCL / Script, and any other required objects).

The entry and modification of the NATCDCSP processing specifications occurs through the use of the **Generate NATCDCSP Objects for Multi File** function. As a pre-requisite to using the **Generate NATCDCSP Objects for Multi File**, the **Production NATCDC SP Process** JCL / Script template must exist; for information on handling this JCL / Script, please refer to the section entitled Configuring JCL / Script.

To generate a NATCDCSP process, perform the following:

1. **Start NatQuery**
   If NatQuery is already started, then insure that NatQuery is on an "open" desktop (I.E. no queries open and no Administrative windows are displayed.

2. **Invoke the Generate NATCDCSP Objects for Multi File function**
   The **Generate NATCDC Object for Multi File** function is accessed by clicking on the **Administer** drop-down menu, then clicking on **NATCDC / NATCDCSP**, and then clicking on **Generate NATCDCSP Objects for Multi File**.

   If the administrator is prevented from accessing this function, then there are two likely causes: Either; no JCL / Script template has been created for the Production NATCDC SP Process template, or no file(s) have been set up to be a **Component of a Multi File Process**. The first situation can be corrected by referring to the previous section entitled Configuring JCL / Script; the second situation can be corrected by referring to the section entitled Generating NATCDC Objects for Single File.

3. **Generate NATCDCSP Objects for Multi File - Select Process Name of Interest**
   In the top of the **Generate NATCDCSP Objects for Multi File** window there is a selection-box labeled "**Select Process Name to Generate**", with this selection box allowing for the selection of a DDM that has been verified for use.

   When the **Generate NATCDC Objects for Single File** is used to generate a **Single Pass Process** type of **Component of Multi File Process**, as part of that specification the

administrator assigned the processing a **Process Name**. The assigned **Process Name** is then used by the **Generate NATCDCSP Objects for Multi File** to collect all processing that has the same **Process Name**, and then generate the processing required to handle all of those files in a single pass.

To utilize this function, the administrator will begin by selecting the **Process Name** of interest.

If the administrator does not find the **Process Name** to be handled by looking at the selections available in the **Select Process Name to Generate** selection box, then the likely cause is that the **Process Name** being searched for was never created through the **Generate NATCDC Objects for Single File** function.

With a **Process Name** selected, the administrator can proceed to the next step.

4. **Generate NATCDCSP Objects for Multi File – Add / Drop Files of Interest**
When a given **Process Name** has been selected, all files that have been handled through the **Generate NATCDC Objects for Single File** function which have been specified as having a **Single Pass Process** type of **Component of Multi File Process** and which have been linked with the selected **Process Name** will be displayed in the **File Encompassed by Selected Process** grid.

   Along with the **File Name** of each file that is linked with the selected **Process Name**, the grid will display each file's **File Number** and whether or not each file is **Included** in the Single Pass Process.

   Using the controls on the File Information tab, the Administrator will make the determination of which files, if any, should be included or excluded from the Single-Pass Multi-File Process about to be generated.

   When the files of interest are selected, the administrator can click on the **File Naming** tab and proceed with the next step.

5. **Generate NATCDCSP Objects for Multi File – File Naming Tab**
The **File Names** tab provides NatQuery with information pertaining to the **File Names Referenced in JCL / Script**, the **File Names used in FTP / Copy**, and the **Name of the Single-Pass Processing Program** that will be generated.

   5.1. **File Names Tab - File Names Referenced in JCL / Script Frame**
   The **File Names Referenced in JCL / Script** fields are the values of the file names that will be used in a Stand Alone Process for substitution into the **Production NATCDC SP Process** JCL / Script template. For important information concerning the use of these files names, please see the **Note'**s found later in this step.

Essentially, the file names available on this frame will be directly substituted into corresponding Dynamic Variables that reside in the **Production NATCDC SP Process** JCL / Script template, thereby allowing that generic template to become specific to the file processing required.

The file names displayed in the **File Names Referenced in JCL / Script** frame will either come from the last values entered into this form for the specified file (if the specified **Process Name** has been successfully processed through the **Generate NATCDCSP Objects for Multi File** function previously), or they can come from internally supplied defaults (that are sensitive to target environment).

Specific information on each of the fields in the **File Names Referenced in JCL / Script** is described later in this step**.**

NOTE #1 – Initial Naming:
In most instances, and particularly with the initial use(s) by the Administrator of the **Generate NATCDCSP Objects for Multi File** function, it is advisable that the administrator not become too consumed with the File naming being used. Once the final JCL / Script is generated and executed, the administrator will better understand how these respective files are used in the final JCL / Script, and will therefore have a more solid understanding of how file naming operates.

Descriptions of File Names
A description of how each of the files names that appear in the **File Names Referenced in JCL / Script** frame are as follows:

5.1.1. **PLOG Data (Input)**

This file name references the PLOG Data file that is output from the initial ADABAS PLOG utility processing. For mainframes, this is the name of the file which contains the output of ADASEL (or the output of ADACDC), and for Open Systems this would be the output of ADAPLP.

In JCL / Script templates, this field equates to the Dynamic Substitution Variable **&&ADASEL-OUTPUT**.

NOTE #1:
At the time of initial generation for a specific file, this file may not physically exist yet, and would not exist until the appropriate PLOG utility program(s) is run.

NOTE #2:
The use of the name **&&ADASEL-OUTPUT** will be the name used for dynamic substitution even though the PLOG was created on an Open System and is then

processed through the ADAPLP utility.

5.1.2. **NATCDCSP Parameters (Input)**

This file name references the Parameter File that will be generated for the specified **Process Name**, with this Parameter File instructing the NATCDCSP process on how to handle the mapping of each requested file from a PLOG source.

In JCL / Script templates, this field equates to the Dynamic Substitution Variable **&&CDC-PARMS**.

5.1.3. **Intermediate File #1 (Temp)**

This file name references a temporary file that is only used to contain transactional information between processing steps.

In JCL / Script templates, this field equates to the dynamic substitution variable **&&CDC-I-DSNAME1**.

5.1.4. **Intermediate File #2 (Temp)**

This file name references a temporary file that is only used to contain transactional information between processing steps in the JCL / Script.

In JCL / Script templates, this field equates to the dynamic substitution variable **&&CDC-I-DSNAME2**.

5.1.5. **Final Data (Output)**

This file name references the primary output file of a NATCDCSP process and is the data file that is created from this process.

In JCL / Script templates, this field equates to the dynamic substitution variable **&&CDC-F-DSNAME**.

5.2. **File Names Tab - File Names to be used in FTP/Copy Frame**

The **File Names to be used in FTP/Copy** fields are the values (or prefixes) that will be applied to the file names of the generated JCL / Script and the generated Parameter File when these files are placed onto the server by automated FTP or File Copy processing.

When creating PLOG processing against a mainframe environment, then in almost all cases the file names entered into these two fields will be identical to the names of the files in 5.1 above, with these fields allowing the JCL file and the Parameter file to be moved under different references. This would then handle the situation where FTP against a server establishes a working directory by default, and the administrator wishes to have the files moved into a different directory.

NOTE (Script Generation):
When handling the generation of Script against Open Systems, NatWorks follows the convention suggested by Software AG for Batch execution, meaning that in addition to a base Script being generated, additional files will be generated that support that Script: Specifically a COMBJIN and a CMSYNIN file will be generated. The base script will refer to these files, and these files will be moved together with the base Script onto the server platform if the software is allowed to. Therefore the field labeled **NATCDCSP Script File Prefix** will be used as the file name prefix for the Script as well as the CMOBJIN and CMSYNIN files, with "sh" or "bat" file extension for the Script, and "txt" extensions for the CMOBJIN and CMSYNIN files.

5.3. **File Names Tab – Post-NATCDCSP Program Name Field**
Subsequent to the PLOG processing that will occur by NATCDCSP, the data needs to be processed by generated NATURAL programs so that all remaining processing of the PLOG data can be completed. The **Post-NATCDCSP Program Name** field provides the name of this program.

By default, NatQuery will provide a default name for the **Post-NATCDCSP Program Name**, with this name beginning with a hard-coded value of "SPP" followed by a NatQuery assigned unique 5-digit number.

The program being named with this field will be the primary NATURAL program executed following a NATCDCSP execution, with this program serving as a "Traffic Program" that branches processing to a NATURAL sub-program (generated as a result of using the **Generate NATCDC Objects for Single File** function) to process individual files. When a sub-program receives control it will read from the PLOG input until all PLOG transactions against that given file are processed, at which time the sub-program will return control to the Traffic Program. The Traffic Program may then call other sub-programs as needed, or it may end processing if all PLOG transactions have been processed.

The administrator should review the provided **Post-NATCDC / NATCDCSP Program Name**, and make any changes to it as required.

NOTE:
It should be remembered that when NatQuery assigns the program name, it does so by only considering the names in use by NatQuery for PLOG processing that exist in the NatQuery environment – it currently has no ability to check the use of program names in the remote NATURAL environment, so changes to the program name as well as the default program names should be scrutinized by the administrator for any possible issues.

6. **Generate NATCDCSP Objects for Multi File – Transactions Tab**
   The **Transactions** tab is used when PLOG processing is being generated against a mainframe environment where JCL needs to reference the Disk Space that will be needed to process PLOG transactions. When a NATCDCSP process is being generated, then the value entered into the **Estimated Number of Transactions** will be used to calculate the allocations needed for Disk Space references in JCL.

   NOTE:
   When PLOG Processing is generated against UNIX, Linux or Windows platforms then the **Transactions** tab will be disabled as this information will not be needed.

7. **Generate NATCDCSP Objects for Multi File – Wrap Up**
   When the information on all appropriate tabs has been reviewed and modified as needed, the administrator can now click the **OK** button.

   This initial result from clicking **OK** will be the invocation of a number of edits against the supplied specifications, and any identified error must be corrected prior to generation and any possible file movement occurring.

   Subsequent prompts and message boxes will depend on the options selected, with window-specific Help being available on any given screen by clicking the **Help** button.

   At some point in the generation process, and depending upon several factors (including the method of integration being used) an administrator may wish to generate all required objects but not introduce these objects into the server environment. The generation process allows for this, and will prompt the user for confirmation on moving generated objects to the server.

   When all editing has been accomplished and all generated objects have been created, the **Generate NATCDCSP Objects for a Multi File** will conclude it's processing with the generation of a report that details what was accomplished. While being presented to the user via a scrollable text window, this report can be printed and will also be saved to disk.

Results of Generation

What is generated by the **Generate NATCDCSP Objects for Multi File** function will depend on several factors.

Unless otherwise specified, the files that are generated into the workstation environment will be placed into the currently specified **Data Download \ Output Directory** of NatQuery (as set through the **NatQuery Configuration** function, **Environment Paths** tab).

The objects that can be generated are:

- Natural Program
- Parameter File (external)
- Parameter File (internal)
- JCL / Script
- Report

Natural Program

If generation is allowed to occur, then besides any other generation that may occur, a NATURAL program will be generated that will be the main processing program behind the execution of the NATCDCSP module. This program will in turn issue CALLNATs to the appropriate individual sub-programs that were generated individually as a result of using the **Generate NATCDC Objects for Single File**.

The program will be generated into the **Data Download \ Output Directory** and will have a file name equal to the value of the **Post-NATCDCSP Program Name** (as specified on the **File Names** tab) with a "**NSP**" extension.

Parameter File (External)

If generation is allowed to occur, then besides any other generation that may occur, an "external" **Parameter** file will be generated that will be used as input to the NATCDCSP process, with the purpose of this parameter file being to described the required field mapping to the NATCDCSP processing module.

This Parameter file will be a single file that is a collection of all parameter files that were generated individually through the use of the **Generate NATCDC Objects for Single File** and which have been included in the NATCDCSP process.

The **Parameter** file will have a file name equal to the value of the **Post-NATCDCSP Program Name** (as specified on the **File Names** tab) with a "**CDC**" file extension.

Parameter File (Internal)

If generation is allowed to occur, then besides any other generation that may occur, an "internal" **Parameter** file will be generated that will be used to save the user's specifications for the currently selected Process Name for subsequent re-presentation to the user when / if the **Process Name** is next selected through the **Generate NATCDCSP Objects for Multi File** function.

The **Parameter** file will have a file name equal to the value of the **Process Name** with a "SPP" file extension.

JCL / Script

If generation is allowed to occur, then besides any other generation that may occur, a JCL or Script file will be generated.

Whether JCL is generated or whether Script is generated depends on the setting of **Server Type**, as specified on the **Connection Info** tab of the **Administer Server Connection Information** function. From an empty desktop, this function is accessed by first clicking on the **Administer** drop-down menu, then clicking on **Environment Configuration**, **Server Connection Configuration**, and then clicking on **Server Information**. If **Server Type** is set to **MVS** or

VSE, then Job Control Language (JCL) will be generated; if the **Server Type** is set to **UNIX** or **NT** then Script will be generated.

The JCL / Script will be generated into the **Data Download \ Output Directory** and will have a file name equal to the value of the **Post-NATCDCSP Program Name** (as specified on the **File Names** tab) with a "JCL" file extension (if Server Type is **MVS** or **VSE**), "bat" (if **Server Type** is **Windows** or "sh" if **UNIX** or **Linux**.

NOTE #1 (Script Generation):
When generating Script for Open Systems, NatWorks products follow the recommendations of Software AG, meaning that in addition to a base Script being generated, additional supporting files are also generated, specifically CMOBJIN and CMSYNIN files. These file will be generated in parallel with the generation of the base script, the script will reference these files, and all of these files will be moved onto the server by NatQuery if the software is instructed to do so.

NOTE #2 (Script Generation):
When handling the generation of Script, the **Production NATCDC SP Process** template will be a single file that actually contains the entire contents of not just the Script, but also the CMOBJIN file, the CMSYNIN file, and the NATURAL Program as well. If FTP or File Copy is allowed to occur, then this single file will be split out into its separate component files and these separate files will physically be created on the server platform. These "split out" component files will not be seen in the **Data Download / Output Directory** – however these can be seen in the path specified as the **Environment Path**.

Report

If generation is allowed to occur, then besides any other generation that may occur, a report file will be created, with this report being the same information displayed at the completion of the **Generated NATCDCSP Objects for Multi File** function.

The Report will be generated into the **Data Download \ Output Directory** and will have a file name equal to the value of the **Post-NATCDCSP Program Name** (as specified on the **File Names** tab) with a "RPT" file extension.

Generating ADABAS PLOG Utility Parameter Cards

In order to utilize the ADABAS utilities that pre-process PLOG files (I.E., select PLOG records from specific files and decompress these records), these utilities need Parameter Cards. In some cases they need very specific Parameters Cards. Providing incorrect Parameter Cards will cause unpredictable results with NATCDC or NATCDCSP when the PLOG transactions are subsequently processed.

To assist the administrator in creating the appropriate parameters to use with the ADASEL, ADACDC, or ADAPLP utilities, NatQuery provides the ability to generate these parameter cards, with this ability embedded into a function called **Generate ADABAS PLOG Utility Parameters**.

To use the **Generate ADABAS PLOG Utility Parameters** function, perform the following steps:

1. **Start NatQuery**
   If NatQuery is already started, then insure that NatQuery is on an "open" desktop (I.E. no queries open and no Administrative windows are displayed.

2. **Invoke the Generate ADABAS PLOG Utility Parameters function**
   The **Generate ADABAS PLOG Utility Parameters** function is accessed by clicking on the **Administer** drop-down menu, then clicking on **NATCDC / NATCDCSP**, and then clicking on **Generate ADABAS PLOG Utility Parameters**.

3. **Generate ADABAS PLOG Utility Parameters – Select Database ID**
   The **Generate ADABAS PLOG Utility Parameters** generates the parameters required based on the Database ID (DBID) of the source ADABAS Database.

   Using the selection box labeled **Database ID to Generate Parameters for**, the administrator will select the appropriate Database ID - individual files were associated with a given DBID through the use of the **Generates NATCDC Objects for Single File** function.

   When a Database ID is selected, the grid control in the lower center of the window will be populated will all source files that have been handled through the **Generate NATCDC Objects for Single File** function that are associated with that DBID.

4. **Generate ADABAS PLOG Utility Parameters – ADABAS Utility Options Frame**
   The **Generate ADABAS PLOG Utility Parameters** function has the ability to generate the parameters for ADASEL, ADACDC, or ADAPLP, and the frame entitled **ADABAS Utility Options** allows the appropriate utility to be selected. These selections will be sensitive to the ADABAS source platform, with defaults provided.

In the case of mainframes, there are two PLOG utilities available, ADASEL or ADACDC. At the present time it is NatWorks' recommendation that the ADASEL utility be utilized to pre-process PLOG data in a mainframe environment over the use of ADACDC. Due to the differences in the output of ADASEL versus ADACDC, and due to the NatWorks recommendation of using ADASEL over ADACDC, the default NATCDC module provided by NatWorks is designed to only support the output of ADASEL. Support for ADACDC is available from NatWorks upon request; please contact NatWorks for further details.

If the ADABAS source database resides on a mainframe and the ADASEL option is selected, then there will be two sub-options available: **LOGINFO** and **EXTENDED**. Except when NatQuery is being used to support NATCDC versions prior to 3.2.0, NatWorks suggests and recommends that the **EXTENDED** option be used. These two options affect the layout of PLOG record "headers" that are output with transactional data, and since the ADASEL parameter **EXTENDED** provides more transactional information than that provided with **LOGINFO**, NatWorks recommends using the **EXTENDED** parameter with ADASEL. All current versions of NATCDC / NATCDCSP will expect the header format produced by the EXTENDED parameter.

In the case of Open System sources such as UNIX, Linux, or NT – there is only the ADAPLP utility available to process PLOGs at this time.

NOTE #1 (Mainframes):
Versions of NATCDC prior to 3.2.0 will only support ADASEL with LOGINFO. Versions of NATCDC 3.2.0 and higher are designed to only support ADASEL / EXTENDED with support for ADASEL / LOGINFO being phased out. Support for ADACDC is available at this time only for a special version of NATCDC 3.2.0.

NATCDCSP 4.2.0 (the initial release of this software) and above is only designed to support ADASEL / EXTENDED, with no support for ADASEL / LOGINFO planned.

NOTE #2 (Mainframes):
One of the capabilities of the ADASEL utility is to "split out" the transactions from individual ADABAS source files into separate files – however ADASEL has a limit of only splitting out / creating 20 files at a time. Since both NATCDC and NATCDCSP have the ability to "select" records of interest from the ADASEL output – there is no need to have ADASEL do anything other than write all records to the same output file (EXPA*n* file).

NOTE #3 (Open Systems):
In contrast to the PLOG utilities available on the mainframe, the ADAPLP utility of ADABAS on Open Systems is rather limited in what it can do. For example: It can only support selecting a single file or all files, it cannot select a specific list of files. Beyond this – it was designed to only produce reports, it was not designed to produce both data files OR reports as the ADASEL utility can. These shortcomings however are handled by a

NatWorks-provided utility program called NATPLP, which essentially operates by converting the output of ADAPLP into a data file that has the same layout of the output of the mainframe utility ADASEL / EXTENDED. Therefore NATPLP will always utilize the output of ADAPLP.

Versions of NATCDC from 3.2.0 and above as well as NATCDCSP 4.2.0 and above supports ADAPLP through NATPLP.

5. **Generate ADABAS PLOG Utility Parameters – File Processing Frame**
Subsequent to selecting the DBID of interest in step 3 above, the grid control located in the File Processing Frame will become populated with all the files that were associated with this DBID value, which was specified by the administrator through the **Generate NATCDC Objects for Single File** function.

   This grid control allows the administrator to be selective about which files should have Parameters generated and which should not; by default all files will be selected for processing (I.E., not omitted from processing).

   Using the controls located in the **File Processing** frame, the administrator should review the files and make any changes as required (**Omit** or not **Omit** Files as necessary).

6. **Generate ADABAS PLOG Utility Parameters – Wrap Up**
When the administrator is satisfied with the file selection, they should click the **OK** button. This will cause NatQuery to internally generate the Parameter Cards needed for the execution of the designated utility.

   Once generated, these parameter cards will be displayed in a report window for review and here they can be cut and pasted as necessary. Additionally, these same Parameter Cards will be written to disk with this output being placed into the designated Data Download / Output directory. This file will be named in the following way:

   CDCnnnnn-z.SEL

   The “CDC” value is a hard-coded prefix, the “nnnnn” is the number of the DBID as selected, the “-“ is a hard-coded hyphen, the “z” is the relative number of the parameter file (used in the situation where ADASEL parameters are being generated and there are more than 20 files to process; meaning that multiple ADASEL parameter files will be created), and “SEL” is the hard-coded file extension.

   Once these cards have been generated, they can then be forwarded to the appropriate DBA responsible for the source ADABAS database for inclusion into the execution of the appropriate ADABAS utility.

NOTE (Mainframes):
When generating the Parameter Cards for the ADASEL utility and in situations where more than 20 ADABAS files are intended to be handled, NatQuery will present a message box that asks if the administrator wishes to generate Parameter Cards that instruct the ADASEL utility to write all processed PLOG transactions to the same file or to separate files. While the administrator may wish to have transactions written to separate files for load balancing or other reasons, NatWorks strongly *suggests* that all transactions should be written to the same file at this time.

Handling Errors with NATCDC / NATCDCSP

By far the most common error that can be encountered with NATCDC or NATCDCSP will involve the use of a DDM that does not exactly match the corresponding FDT for the file being processed.

To understand the potential effects of this situation, an appreciation of how NATCDC / NATCDCSP perform their processing is required.

In its raw form, and setting aside for the moment the fact that PLOG records are written with headers, a PLOG contains transactional records that are in the same form as they are stored in ADABAS, I.E., they are "compressed". When these records are decompressed by the appropriate ADABAS PLOG utility, they will be written as variable length records.

In converting the variable length records to fixed length records, NATCDC and NATCDCSP "maps" the fields in a variable length record into a fixed length record. In order to do that, these modules need to know the exact layout of the file being processed, and this information comes from the DDM that NatQuery is supplied with.

When NATCDC and NATCDCSP maps non-recurring element fields from an input file (provided by an ADABAS PLOG utility) into the output file, it does so in "chunks" of one byte or more, with each "chunk" representing a field as seen in the DDM. As these chunks are moved, NATCDC and NATCDCSP do not do any verification / editing on the actual contents of the field's / chunk's content (for example alpha vs. numeric): To do so would cause a significant amount of overhead and would significantly encumber an otherwise efficient process. It is therefore possible that while NATCDC or NATCDCSP might conclude to a successful End-Of-Job, they may map a file incorrectly because they were given an incorrect DDM. Depending on the format and layout of the fields in the source file, incorrect mapping may or may not be caught by the generated post-NATCDC / NATCDCSP NATURAL programs.

When NATCDC or NATCDCSP map recurring fields from an input file into an output file, these recurring fields will be preceded in the input file by the C* value representing how many actual recurring fields follow. NATCDC and NATCDCSP examine this value, and based on this value in comparison to the number of occurrences requested (as specified via the **Administer Occurrence Information** function of NatQuery), NATCDC and NATCDCSP then output the requested number of occurrences by truncating or padding fields / chunks. When NATCDC or NATCDCSP examine the C* value, they edit this value prior to acting upon it to insure that it is both numeric in content and within acceptable bounds. If this is not the case: NATCDC and NATCDCSP will abort further processing with regard to the file being processed.

Error Resolution

Errors occurring out of a NATCDC or NATCDCSP process will typically involve one of the following areas:

- Errors from ADASEL, ADACDC, or ADAPLP
- Errors from NATCDC or NATCDCSP
- Errors from Generated NATURAL Processing Programs

Errors from ADASEL, ADACDC, or ADAPLP

Error occurring in the ADABAS PLOG utilities should generally be referred to Software AG, as NatWorks has no control over the functioning of these utilities. Prior to contacting Software AG however, it is strongly advised that the Parameter Cards being given the affected utility be examined for correctness.

If a customer would like to discuss any errors that are encountered with the execution of one of the above ADABAS utilities with NatWorks prior to calling Software AG – NatWorks will welcome the call and will assist with troubleshooting to the best of our abilities.

Errors from NATCDC or NATCDCSP

In any situation where NATCDC or NATCDCSP aborts, the ON ERROR processing of these modules will output information that will pinpoint the likely cause of the error. In almost all cases, this situation will be caused by NATCDC or NATCDCSP encountering a field that the DDM indicates is a C* value, but the contents of this field is either not numeric or not within accepted bounds.

This will mean that NATCDC or NATCDCSP were given DDMs that did not exactly match the layout of the corresponding FDT, and the corrective action is to download a new DDM, insure that this DDM exactly matches the corresponding FDT, and then re-generate the NATCDC / NATCDCSP processing involved. The handling of a DDM / FDT is detailed in the section entitled Overview of Configuring DDMs / FDTs.

In situations where NATCDCSP aborts due to a mapping issue, another possible cause is that the SORT that precedes the NATCDCSP execution did not properly handle a variable record-length SORT. If the SORT is not properly instructed that it is sorting a variable-length record, then the 4-byte inclusive record length that precedes the actual data record can be interpreted as data – leading to incorrect mapping.

If NATCDC or NATCDCSP continues to report an error after following the corrective action(s) described above, please contact NatWorks or your NatWorks representative for a resolution.

Errors from Generated NATURAL Processing Programs

In any situation where a generated NATURAL Processing Program aborts, the likely cause of this will be that the processing was given a DDM that does not match the corresponding FDT. As described above, this problem may not be discovered via an execution of NATCDC or NATCDCSP, and only encountered when (for example) a generated processing program attempts to move the contents of a given field with an Edit Mask that is compatible with the expected contents but which is not compatible with the actual contents.

If an error encountered with a generated processing program is in any way related to the movement of a field with an edit mask or the assignment of a field, the first thing that should be checked is whether or not the DDM being used to process the given file actually represents the true file layout or not. The corrective action will therefore be to download a new DDM for the specific file, insure that this DDM exactly matches the corresponding FDT, and then re-generate the NATCDC / NATCDCSP processing involved. The handling of a DDM / FDT is detailed in the section entitled Overview of Configuring DDMs / FDTs.

In situations where a generated Natural Program aborts, another possible cause is that the SORT that follows the NATCDC execution may have not properly handle the variable record-length SORT. If the SORT is not properly instructed that it is sorting a variable-length record, then the 4-byte inclusive record length that precedes the actual data record can be interpreted as data – leading to incorrect mapping.

If a generated NATURAL Processing Program aborts for any other reason, or still aborts after insuring that the DDM does in fact match the corresponding FDT and that there is no fault with the SORT: The problem may be immediately addressed by manually correcting the generated NATURAL code. While this can possible solve the issue at hand: NatWorks should be notified of the error to allow for internal corrective action to occur if needed.

Handling NATCDC / NATCDCSP Modifications

Modifications to an existing NATCDC / NATCDCSP process can take many forms. The most commonly anticipated changes, and the appropriate actions to take for these changes, are discussed below.

Handling Physical Changes to a ADABAS File

The following steps should be followed to implement a new NATCDC / NATCDCSP process when a change to an existing ADABAS file has occurred (new fields added or existing fields removed).

1. **Insure that a DDM is generated from the new ADABAS FDT**
   A DDM should be generated directly from the FDT for the new file. Alternatively, a correct DDM can be created by reviewing the steps outlined in Overview of Configuring DDMs / FDTs.

2. **Verify the Environment Configuration**
   The Verify Environment Configuration function performs basic edits against all DDMs in a given Environment Configuration to insure that the required administrative tasks have been completed against each DDM.

   If any verification errors occur against the newly downloaded DDM, then the Administrator should perform whatever corrective action is necessary such that the new DDM will pass verification.

3. **Generate NATCDC Objects for Single File**
   Once the new version of the DDM is present in NatQuery, the **Generate NATCDC Object for Single File** should be used to re-generate the processing required for that specific file and any other files that may have changed.

   If the file is a **Component of a Multi-File Process** (as opposed to a **Stand Alone Process** – with both of these options being specified through the **Generate NATCDC Objects for Single File**), then subsequent to using the **Generate NATCDC Objects for Single File**, the **Generate Objects for Multi File** function should be used.

4. **Third-Party Integration**
   If the desired target of a NATCDC / NATCDCSP process is a third-party ETL or Data Warehousing tool such as IBM's Web Sphere Data Integration suite – formerly; Ascential Software's Data Stage product suite, then it is typical that the Generate NATCDC Objects for Single File will create a DSX file or a CFD file that is specific to the given file. The new DSX or CFD should therefore be immediately provided to the

ETL tool so that it can properly respond to the new layout of the CDC file.

If direct integration to Oracle or SQL Server is desired, then the **Generate NATCDC Objects for Single File** will generate the appropriate processing to handle the data into the supported RDBMS target.

5. **Stage new NATCDC / NATCDCSP Processing into Production**
   As a final step to implementing a change, the new NATCDC / NATCDCSP processing should be introduced into the production environment.

Handling Modifications to an Existing Process

In situations where an existing NATCDC / NATCDCSP process needs to be modified, but there have been no changes to the ADABAS file itself, the following steps should be taken to implement the required changes.

1. **Generate NATCDC Objects for Single File**
   Using the Generate NATCDC Objects for Single File function, the user would make the desired changes to the given file's current processing specification.

2. **Generate NATCDCSP Objects for Multi File**
   If the file is designated as a Component of a Multi-File process, then the Generate NATCDCSP Objects for Multi File function should now be used to re-generate the full process that will handle multiple files in a single pass.

3. **Third Party Integration**
   If the desired target of a NATCDC / NATCDCSP process is a third-party ETL or Data Warehousing tool such as IBM's Web Sphere Data Integration suite – formerly; Ascential Software's Data Stage product suite, then the Generate NATCDC Objects for Single File Process will have created either a DSX file or a CFD file. These newly generated files should then be provided to the ETL software so that it can properly respond to the new layout of the CDC file.

   If direct integration to Oracle or SQL Server is desired, then the **Generate NATCDC Objects for Single File** will generate the appropriate processing to handle the data into the supported RDBMS target.

4. **Stage new NATCDC / NATCDCSP Process into Production**
   As a final step to implementing a change, the new NATCDC / NATCDCSP processing should be introduced into the production environment.

NATCDC / NATCDCSP Production Operation

When ADABAS is run in a production environment, it is typical that the Protection Logging (PLOG) feature is enabled so that transactional changes can be captured and used later in the event of a Recovery operation. While ADABAS can be run without the benefit of PLOG processing, it is a requirement for NATCDC and NATCDCSP that PLOG processing be operational. It is a further requirement for NATCDC / NATCDCSP that the ADABAS utility ADASEL be available to process the PLOG output in mainframes environments, and ADAPLP be available to process PLOG output in Open Systems Environments.

Protection Logging can be set up for ADABAS in one of two ways, either Protection Logs can be written immediately to a sequential dataset (DDSIBA or SIBA), or "Dual-Logging" can be utilized wherein two disk datasets are utilized. Under Dual Logging, one disk dataset is "active" (being written to) in order to capture transactional changes. When this first disk area fills up, ADABAS automatically switches to write transactional changes to the second disk dataset. When this switch occurs, it is normal operating procedure within ADABAS to utilize User Exit 2 to launch a batch job that will copy the contents of the previously active PLOG dataset to tape.

While PLOG switching is done automatically by ADABAS at appropriate times, a PLOG switch can also be accomplished upon request through the use of the OPERCOM utility in conjunction with the FEOFPL command.

Processing PLOGs

As has been indicated previously, NATCDC and NATCDSP are dependent upon the execution of the ADASEL or ADAPLP utility (used with NATPLP), with both of these utilities dependent upon PLOG files. Therefore, beyond the capture of data into a PLOG, a procedure or procedures must be implemented that will allow the processing of PLOG information through the appropriate ADABAS utility (ADASEL or ADAPLP & NATPLP).

While NATCDC / NATCDCSP addresses the processing of data provided from an execution of the ADABAS PLOG utility, the manner and scheduling by which generated PLOG records are processed through the ADABAS utility, and the manner and scheduling of how NATCDC / NATCDCSP generated processes are executed is left up to the individual customer. This is due to the fact that the requirements for when NATCDC / NATCDCSP output is needed can vary significantly from one customer environment to the next.

PLOG Processing Considerations

In considering what the best approach for implementing the processing of PLOG data at a given site, the NATCDC / NATCDCSP Administrator and a Data Base Administrator should consider the timing of when the ADABAS PLOG utility process can be executed separately from when the subsequent NATCDC / NATCDCSP processes are run. For instance, it is possible to run the ADABAS utility as part of a PLOG switch such that processed PLOG data is processed almost immediately (appending or MODing data into an output file), while the NATCDC / NATCDCSP processing of this data may only have to run nightly or weekly.

Of course, it is also possible to run all processing (ADABAS utility followed by NATCDC or NATCDCSP processing) as part of a PLOG switch – thus approaching a real time feed of data.

NATCDC / NATCDCSP Processing Approaches

The following approaches should be considered as *suggestions* for installing PLOG processing:

- **Approach #1**
  As the most direct approach to implementing the required PLOG processing, it is suggested that the JCL / Script that is submitted automatically by ADABAS to copy a PLOG dataset be modified so that this job either launches or additionally executes the required ADABAS utility and / or NATCDC / NATCDCSP processing.

  If your site uses a JCL / Script scheduling package, then one implementation approach would be to have the PLOG copy job launch the appropriate PLOG processing.

  Another implementation approach would be to have the PLOG copy job execute a batch NATURAL job step, and in this job step have NATURAL execute a Natural program that then submits the required PLOG processing. If you would like to pursue this approach, NatWorks can provide you with a program example that will perform a JCL submission through NATRJE upon request.

  Still another suggestion might be to modify the PLOG copy job so that this job directly handles the ADABAS utility execution and the NATCDC / NATCDCSP processing.

  When considering these approaches, it must be understood that it is highly desirable to NOT encumber the ADABAS submitted ADARES PLCOPY job unduly – as ADABAS will look for the successful completion of this job in order to know when a subsequent PLOG switch can occur without contention. It should also be understood that ADABAS processing can cease in a situation where dual-logging is employed; one PLOG dataset fills up and is in the process of being copied / processed, and during the time the second PLOG dataset also fills up.

- **Approach #2**
  A customized user process is employed that will dynamically provide the appropriate PLOG copy tapes for execution through the required ADABAS utility and NATCDC / NATCDCSP processes.

  One approach for this on mainframes would the use of a LISTCAT operation against the generational PLOG copy tapes with this information then written out to a specified disk area. Using a Natural program and a separate “logging” file, the LISTCAT data combined with the Logging data can be used to dynamically create a JCL stream to pick up the required datasets for an ADASEL process.

Please feel free to contact NatWorks, Inc. for assistance in discussing or implementing the required PLOG processing of NATCDC / NATCDCSP. Information on contacting NatWorks, Inc. can be found in the section entitled Contacting NatWorks.

Handling File Changes with PLOG Processing

When a file is defined to ADABAS, a File Definition Table (FDT) is created that explicitly defines the layout of that file. This FDT is then physically stored in the File Control Block (FCB) of the specific ADABAS database that the given file is contained in.

Beyond defining the physical layout of a given ADABAS file, an FDT is used by ADABAS to support the automatic Compression processing that is applied to all records that are internally stored into any given ADABAS file. Conversely: When ADABAS Decompresses a record, the FDT of the file is again utilized to properly decompress the record.

While the exact manner in which ADABAS performs its compression and decompression is well outside the scope of this document, it should be generally understood that the FDT of a file controls how ADABAS will compress any given ADABAS record, and it also controls how ADABAS will decompress a compressed record.

With the above understood, three points must now be considered by any organization looking to process an ADABAS PLOG:

1. **PLOG Records are Stored into the PLOG in Compressed Format**.
   The transactional records that a PLOG contains are in the same compressed format as what ADABAS stores internally. This means that records written to a PLOG must at some point be decompressed in order to have PLOG record contents be usable.

2. **ADABAS Does Not "Version" FDTs**.
   ADABAS does not inherently provide any versioning mechanism for FDTs; at any given time there is only one version of a file's FDT available to a given ADABAS database, and that is the "current" version of the given file's FDT.

3. **ADABAS PLOG Utilities Require an FDT to Decompress Records**.
   Since one of the primary functions of ADASEL, ADACDC, or ADAPLP in PLOG processing is to Decompress specific PLOG records, these utilities must utilize the file's FDT to accomplish this decompression. This FDT is resolved from the ADABAS database that the utility is pointed at.

Assuming that any given ADABAS file never changes, there will never be an issue with the PLOG processing against that specific file. This would be because the ADABAS PLOG utility

would always reference the correct FDT, and therefore the PLOG records would always be decompressed in the same manner.

However, if the FDT for any given file ever does change, then a unique situation arises:

> ***At the point in time that the FDT of the file is physically changed in an ADABAS database's FCB, if an ADABAS PLOG utility is then pointed at that database, the ADABAS PLOG utility will only be able to "see" the current version of the FDT. What this means is: When an ADABAS PLOG utility is run against a PLOG to process transactions against a given file, the FDT for that file should be the same as when the PLOG was created. If the FDT has changed since the PLOG was created, then depending upon the nature of the change: The ADABAS PLOG utility will likely fail to properly decompress the file's transactional PLOG records.***

While the above situation does pose an issue for proper PLOG processing across FDT changes, there are several ways to insure continued PLOG processing, including:

1. **Handling FDT Changes with Precise Scheduling**

2. **Handling FDT Changes by "Versioning" FDTs**

3. **Full Re-Extraction Followed by "New" PLOG Processing**

While the handling of each of the above methods will be described in following sections, NatWorks makes the following suggestions to assist a NATCDC / NATCDCSP customer in determining the best method of handling ongoing PLOG processing in conjunction with FDT changes:

1. If the only use of PLOG processing is for ADABAS Change Data Capture, then Option #1 represents the fastest method of handling an FDT change, Option #2 represents the most work to set up but is the best approach overall, and Option #3 represents the easiest approach but is also generally the most costly in terms of CPU.

2. If the use of PLOG processing will be for ADABAS Auditing but auditing will only occur against "current" versions of FDTs, then there is really no action required.

3. If the use of PLOG processing will be for ADABAS Auditing and this auditing must handle both current FDTs and historical FDTs, then option #2 is the best solution.

4. If at any time customers find themselves in a situation where a PLOG contains records that were stored under an FDT that was not “versioned”, then the only choice is option #3.

Handling FDT Changes with Precise Scheduling

This approach utilizes precise timing so that all PLOG processing against a “current” version of an FDT is completed prior to physically installing a “new” FDT and then launching a “new” PLOG process.

This approach would be handled as follows:

1. Lock the ADABAS file to prevent further changes. Locking a file is a fairly standard practice for a file that is about to be changed, and allows the DBA to perform the proper steps to institute a new FDT / ADABAS File.

2. While the File of interest is locked, and prior to the DBA beginning to install the changed FDT / file, a PLOG switch is forced. This PLOG would then contain the “final” changes against the file under the “current” FDT.

3. While the File of interest is still locked, and prior to the DBA beginning to install the changed FDT and file, all outstanding PLOG datasets are run through the existing NATCDC / NATCDCSP PLOG processing.

4. Once all outstanding PLOG processing are completed, the DBA can then initiate whatever steps are required to implement the FDT change.

5. Once the “new” version of the FDT / DDM is available, this DDM / FDT needs to be provided to NatQuery and then a “new” NATCDC / NATCDCSP PLOG process can be generated.

6. The “new” NATCDC PLOG process is then used for all subsequent PLOG processing for the file of interest until the next file change.

By using the above approach, the ADABAS utility will always use the correct FDT to decompress a PLOG dataset, and issues with FDT changes over time can be avoided.

Handling File Changes By "Versioning" FDTs

This approach essentially creates "dummy" ADABAS databases for the sole purpose of allowing FDTs that are active at a given point in time to become "versioned". These "dummy" databases will ***not*** be used to contain any actual business data. These "dummy" databases will only be used to capture the FCB and FDT information necessary to allow the ADABAS utility to correctly interpret the PLOG data being processed at a later point in time.

Creating a FDT Version Database (Mainframes)

Prior to implementing any FDT change to an ADABAS database from which PLOGS will be subsequently processed, perform the following steps to create an FDT Version database:

1. **Define and format a database for the FDT Version Database**.
   NatWorks recommends that an FDT Version Database be defined using a 2-byte DBID. The DBID should be a reserved number or range of numbers that allow for the full separation of "live" ADABAS databases from their FDT Version Database counterparts, but still retaining a method of "logically" associating these databases together. For example, add 1000 to the production DBID to arrive at the DBID number that will be given to an FDT Version Database.

   The space required for and FDT Version Database is minimal, and will be only the space needed define the FDT Version Database itself along with the FCB, and FDT information. The space required should not need to exceed 22 cylinders.

   The following description describes the steps and utilities used to define an FDT Version Database (please refer to the ADABAS Utilities manual for sample JCL / Script, available parameters, and other options for each listed utility):

   1.1 **ADAFRM Utility**
   Run the ADAFRM utility to allocate and format the space required for the FDT Version Database.

   NatWorks suggests the following parameters:

```
ADAFRM ASSOFRM SIZE=200B
ADAFRM DATAFRM SIZE=100B
ADAFRM WORKFRM SIZE=300B
ADAFRM TEMPFRM SIZE=50B
```

   1.2 **ADADEF Utility**
   Run the ADADEF utility to define the database and required checkpoint file.

   NatWorks suggests the following parameters:

```
ADADEF DEFINE
ADADEF DBIDENT=FDT-VERSION-DATABASE-ID
ADADEF DBNAME=FDT-VERSION-DATABASE
ADADEF ASSOSIZE=200B
ADADEF DATASIZE=100B
```

```
ADADEF WORKSIZE=300B
ADADEF MAXFILES=200
*
ADADEF FILE=1,CHECKPOINT
ADADEF NAME='CHECKPOINT',MAXISN=5000,UISIZE=10B
ADADEF DSSIZE=50B,NISIZE=5B,UISIZE=2B
```

1.3 **Copy the FCBs and FDTs into the FDT Version Database**
There are a number of ways to accomplish this task including (but not limited to):

1.3.1 Generate ADACMP cards from Predict and then use these cards as input to the ADABAS ADALOD utility with the LOAD parameter against the FDT Version Database.

1.3.2 Unload the FCB and FDT information from the "live" ADABAS database using the ADABAS ADAULD utility using the following parameters to unload this information:

```
ADAULD MODE=SHORT,NUMREC=0
```

With this information unloaded from the "live" database, this information can now be loaded into the FDT Version Database using the ADABAS ADALOD utility with the LOAD parameter.

1.3.3 Use the Predict-to-AOS Bridge to generate the FDTs directly from Predict into the FDT Version database.

1.4 **Backup the FDT Version database**
Using IEBGENER or a similar utility, copy the newly created FDT Version Database to a Generation Data Group (GDG). The GDG will allow subsequent ADASEL executions to be pointed at a specific FDT Version Database to then support proper PLOG decompression.

1.5 **Verify the FDT Version database**
Use the ADAREP utility against both the defined FDT Version database and the GDG version, and retain this output for subsequent referencing.

The above steps will create an FDT Version Database for a given point in time. By tracking the period of time that a given FDT Version database covers, it will subsequently be possible to run an ADASEL utility against a historical PLOG by pointing the ADASEL utility at the appropriate FDT Version Database that captured the FDTs for that given point in time.

Utilizing a FDT Version database (Mainframes)

With FDT version databases created, PLOG processing against historical FDTs is achieved by running the ADASEL utility against a historical PLOG dataset while pointing the ADASEL utility at the appropriate FDT Version Database that contains the FDTs that were in effect when that PLOG was created.

When the ADASEL utility is run, the JCL that runs the utility should have:

- DDSIIN pointed at the historical PLOG dataset to be processed
- DDASSOR1 will be pointed at the FDT Version Database's ASSO dataset
- DDCARD should minimally contain:

```
ADARUN MODE=SINGLE
ADARUN PROG=ADASEL
ADARUN DB=FDT-VERSION-DATABASE-ID
```

- DDKARTE should have the following general format:

```
SELECT ALL RECORDS FROM FILE nnnnn
     OUTPUT WITH LOGINFO TO EXPAn
END
```

The above will then allow ADASEL to be used against a historical version of a PLOG.

Handling a PLOG that Spans an FDT change

When a PLOG dataset is created and this PLOG spans a period of time in which a FDT change was implemented, and assuming that an FDT Version Database was created that contains the "previous" version of the FDTs, then this PLOG can be properly processed by utilizing the TIME parameter of the ADASEL utility.

The TIME parameter instructs ADASEL to process only those PLOG records for a given span of time. By using this parameter in separate runs of ADASEL, a PLOG that contains both (different) FDT versions of the same file can be successfully processed by pointing ADASEL at the appropriate ADABAS database (either the "live" ADABAS database or one or more historical FDT Version database(s)).

Retention of CDCASSO generations

FDT Version Database need only be kept for as long as an organization retains its PLOGs datasets.

Handling ADABAS Upgrades

NatWorks recommends treating an ADABAS upgrade the same as an FDT change, meaning that just prior to implementing a new version of ADABAS; an FDT Version database is created to capture the FCB and FDTs.

Dynamic Substitution Variable Reference

As described in the following section, and in even greater detail in the NatQuery Installation and Operation manual, NatQuery allows for the definition of “generic” JCL templates.

These “generic” templates are defined as generic through the use of dynamic substitution variables, or variable that NatQuery will dynamically replace with valid values to deliver JCL templates that are unique to a given process.

While there are a great many dynamic variables available for use in the Administer FTP JCL / Script function – many of these variables are specific to NATCDC. The following table outlines these NATCDC specific dynamic variables and details how these variables are initialized.

Dynamic Variable Name	Definition / Initialization
&&CDC-PARMS	This tag is the reference to the name of the NatQuery generated parameter file that is specific to a given ADABAS PLOG file. This file will be used as input to the JCL / Script step that executes the core NATCDC / NATCDCSP object module. This parameter file will be generated by NatQuery, and NatQuery has the ability to move this parameter file onto the NATURAL server. **Notes:** Previously, this tag was known as **&&NATCDC-PARMS**. On Open Systems, the parameter file is also used as input to NATPLP.

&&CDC-ADASEL	This tag is the name of the output file of ADASEL or ADACDC (mainframes), or the combination of ADAPLP and NATPLP (Open Systems). This file will be used as input to a NATCDC or NATCDCSP process. **Notes:** This tag was previously known as **&&NATCDC-ADASEL**. Even Though ADASEL may not be the actual utility employed; this dynamic variable should still be used.
&&CDC-I-DSNAME1	This tag represents the name of one of two "intermediate" files that are needed for a NATCDC or NATCDCSP process. For NATCDC processing, this will be the name of the file output by the execution of NATCDC, which is then used as input to a SORT. For NATCDCSP processing, this will be the name of the file output by a SORT, which is then used as input to NATCDCSP. **Notes:** This tag was previously known as **&&NATCDC-INTRM1**. This file is only required during the processing of NATCDC or NATCDCSP, and can be immediately deleted subsequent to that processing.

&&CDC-I-DSNAME2	This tag represents the name of one of two "intermediate" files that are needed for a NATCDC or NATCDCSP process. For NATCDC processing, this will be the name of the file output by a SORT, which is then used as input to generated NATURAL program processing. For NATCDCSP processing, this will be the name of the file output by NATCDCSP, which is then used as input to generated NATURAL program processing. **Notes:** This tag was previously known as **&&NATCDC-INTRM2**. This file is only required during the processing of NATCDC or NATCDCSP, and can be immediately deleted subsequent to that processing.
&&CDC-I-UNITS	This tag is typically only utilized when the server is a mainframe. The tag will be set to the disk allocation unit (TRACKS or CYLINDERS for VSE, TRK or CYL for MVS) that NATCDC has determined is suitable for the Interim disk requirements (I.E. the disk requirements for &&CDC-I-DSNAME1 and &&CDC-I-DSNAME2). **Notes:** This tag was previously known as **&&NATCDC-INTRM-UNITS**.

&&CDC-I-PRIMARY	This tag is typically only utilized when the server is a mainframe. The tag will be set to the number of primary disk allocation units determined suitable for the Interim disk requirements (I.E. the number of primary disk allocation units of &&CDC-I-UNITS). **Notes:** This tag was previously known as **&&NATCDC-INTRM-PRIMARY**.
&&CDC-I-2NDDARY	This tag is typically only utilized when the server is a mainframe. The tag will be set to the number of secondary disk allocation units determined suitable for the Interim disk requirements (I.E. the number of secondary disk allocation units of &&CDC-I-UNITS). **Notes:** This tag was previously known as **&&NATCDC-INTRM-SECONDARY**.
&&CDC-I-LRECL	This tag is typically only utilized when the server is a mainframe. This tag will be set to the Logical Record Length (LRECL) that NATCDC has determined is suitable for the LRECL of the Interim files (I.E. the LRECL for &&CDC-I-DSNAME1 and &&CDC-I-DSNAME2). **Notes:** This tag was previously known as **&&NATCDC-INTRM-LRECL**.

&&CDC-I-BLKSIZE	This tag is typically only utilized when the server is a mainframe. This tag will be set to the Block Size (BLKSIZE) that NATCDC / NATCDCSP has determined suitable for the Interim LRECL files (I.E. the LRECL for &&CDC-I-DSNAME1 and &&CDC-I-DSNAME2). **Notes:** This tag was previously known as **&&NATCDC-INTRM-BLKSIZE**.
&&CDC-S-LRECL	This tag will be calculated to be the value of &&CDC-I-LRECL + 4, such that system SORT programs can properly reference a variable-length record (which has the inclusive record length in the first 4 bytes).
&&CDC-F-DSNAME	This tag is the name of the file that will be created by the execution of the NATCDC process, and will contain the primary output of a NATCDC / NATCDCSP process, or the final data requested by the user. **Notes:** This tag was previously known as **&&NATCDC-FINAL**.
&&CDC-F-UNITS	This tag is typically only utilized when the server is a mainframe. The tag will be set to the disk allocation unit (TRACKS or CYLINDERS for VSE, TRK or CYL for MVS) that NATCDC has determined suitable for the Final disk requirements (I.E. the disk requirements for &&CDC-F-DSNAME). **Notes:** This tag was previously known as **&&NATCDC-FINAL-UNITS**.

&&CDC-F-PRIMARY	This tag is typically only utilized when the server is a mainframe. The tag will be set to the number of primary disk allocation units determined suitable for the Final disk requirements (I.E. the number of primary disk allocation units of &&CDC-F-UNITS). **Notes:** This tag was previously known as **&&NATCDC-FINAL-PRIMARY**.
&&CDC-F-2NDARY	This tag is typically only utilized when the server is a mainframe. The tag will be set to the number of secondary disk allocation units determined suitable for the Final disk requirements (I.E. the number of secondary disk allocation units of &&CDC-F-UNITS). **Notes:** This tag was previously known as **&&NATCDC-FINAL-SECONDARY**.
&&CDC-F-LRECL	This tag is typically only utilized when the server is a mainframe. This tag will be set to the Logical Record Length (LRECL) that NATCDC has determined is suitable for the LRECL of the Final file (I.E. the LRECL for &&CDC-F-DSNAME). **Notes:** This tag was previously known as **&&NATCDC-FINAL-LRECL**.

&&CDC-F-BLKSIZE	This tag is typically only utilized when the server is a mainframe. This tag will be set to the Block Size (BLKSIZE) that NATCDC has determined suitable for the LRECL if the Final file (I.E. the LRECL for &&CDC-F-DSNAME). **Notes:** This tag was previously known as **&&NATCDC-FINAL-BLKSIZE**.
&&CDC-TOTALS	This tag is utilized to provide the dataset / file name of the optional Totals file that can be optionally produced by NATCDC.

Contacting NatWorks

NatWorks realizes that any person who will install, administer and use NatQuery or NATCDC / NATCDCSP may require assistance from our staff. We are pleased to provide the following methods of contacting us.

Web Site

When looking for general information or to download the latest versions of NatQuery or NATCDC / NATCDCSP, the reader should refer to the NatWorks, Inc. website located at URL **http://www.natworks-inc.com**.

Phone Support

If you are experiencing a problem with NatQuery, NATCDC, or NATCDCSP, or have questions or concerns that require immediate attention, you may call NatWorks during normal business hours from 9:00 AM to 5:30 PM, Eastern Standard Time (EST), at (802) 485-6112.

E-Mail Support

If you are experiencing a problem with NatQuery, NATCDC, or NATCDCSP, or have questions or concerns that do not require immediate attention, then you can contact NatWorks, Inc. through e-mail using the following:

- Technical Support – techsupport@natworks-inc.com
- General Information – info@natworks-inc.com

Thank you!