Natural Engineer

Concepts and Facilities

Version 9.1

October 2018

ADABAS & NATURAL

Manual Order Number: NEE91-006ALL

This document applies to Natural Engineer version 9.1 and to all subsequent releases.

Specifications contained herein are subject to change, and these changes will be reported in subsequent revisions or editions.

Readers' comments are welcomed. Comments may be addressed to the Documentation Department at the address on the back cover. Internet users may send comments to the following e-mail address:

document@gensystems.com

Acknowledgements

TABLE OF CONTENTS

ABOUT THIS MANUAL

Purpose of this manual

This manual contains the Concepts and Facilities for Natural Engineer.

It describes some of the concepts and principles of operation in using Natural Engineer.

Target Audience

The target audience for this manual is intended to be any User of Natural Engineer at any level of experience.

Typographical Conventions used in this manual

The following conventions are used throughout this manual:

UPPERCASE TIMES	Commands, statements, names of programs and utilities referred to in text paragraphs appear in normal (Times) uppercase.
`UPPERCASE BOLD COURIER`	In illustrations or examples of commands, items in uppercase bold courier must be typed in as they appear.
< >	Items in angled brackets are placeholders for user-supplied information. For example, if asked to enter <file number>, you must type the number of the required file.
<u>Underlined</u>	Underlined parts of text are hyperlinks to other parts within the online source manual. This manual was written in MS-Word 97 using the "hyperlink" feature.

The following symbols are used for instructions:

⇒	Marks the beginning of an instruction set.
☐	Indicates that the instruction set consists of a single step.
1.	Indicates the first of a number of steps.

How this manual is organized

This manual is organized to reflect all the Concepts & Facilities of Natural Engineer in the following chapters:

Chapter	Contents
1	Provides an overview of the main Natural Engineer processes.
2	Describes some of the common facilities available within Natural Engineer.
3	Describes the User Exits and Application Programming Interfaces available within Natural Engineer.

Terminology

This section offers some of the terms that are specific to the Natural Engineer product.

Note: Familiarity is assumed with the general terminology of Natural, Adabas, Microsoft and Mainframe operating systems.

Analysis

The Analysis process of Natural Engineer searches application data within the Natural Engineer Repository, according to specified Search Criteria and generates reports on the search results.

Application

An Application is a library or group of related libraries, which define a complete Application. In Natural Engineer, the Application can have a one-to-one relationship with a single library of the same name, or a library of a different name, as well as related steplibs. The Application refers to all the source code from these libraries, which Natural Engineer loads into the Repository.

Browser

An Internet Browser such as Microsoft Internet Explorer or Netscape.

Category

Categories in Natural Engineer specify whether and how a Modification is applied to the Natural code. Valid categories are: Automatic change, Manual change, Reject the default Modification, No change to the data item, and the data item is in Generated Code.

A category is further broken down according to type of change (for example: Keyword, Literal, Data Item, Database Access, Definition).

Cobol

Abbreviation of Common Business Orientated Language. A programming language.

Cobol Link

A Cobol Link is the link between the individual Cobol modules and the executable Cobol program referenced in the JCL object.

Consistency

An option in the Analysis process that causes Natural Engineer to trace an Impact through the code, using left and right argument resolution to identify further code impacted by the code found.

Database Access Definition

A collective term used to identify DDMs, SQL Tables or Predict User Views.

Data Item

A collective term used for any data fields within a programming object. These can be user-defined variables, DDM fields or System Variables. It is inter-changeable with the term ‘variable’.

Environment

The Environment process is the means by which Natural Engineer generates a structured view of the application code in the Natural Engineer Repository. This provides application analysis reports and inventory information on the application and is used as the basis for Impact Analysis.

Exception

An Exception is an Item identified as impacted that does not require a Modification. Where there are a few similar Exception Items, they can be treated as Exceptions, and rejected in the Modification review process. Where there are many similar (therefore not Exceptions), consideration should be given to changing the Search Criteria so they are not identified as impacted in the first place.

Generated Code

This is code which has been generated by a Natural code generator, such as Construct, and which is not normally modified directly in the Natural editor.

Impact

An Impact is an instance of a Natural code Item; e.g., data item or statement (a “hit” scored by the Analysis process) that matches the defined Search Criteria used in the Analysis process.

Iteration

An Iteration is one examination cycle of a field identified according to the specified Search Criteria. For example, one Iteration is reading the field right to left. Multiple Iterations are performed when the option of ‘Consistency’ or Multi Search is requested for Analysis, and Natural Engineer performs as many Iterations as necessary to exhaust all possibilities of expressing and tracing the field, and can be limited by a setting in the NATENG.INI file.

JCL

Job Control Language.

JCL object

A JCL object is a collection of Job Control statements in the order which they are to be executed in a mainframe batch environment. Commonly referred to as JCL.

Library

A single library of source code, which exists in the Natural system file.

Modification

A Modification is a change suggested or made to an object or data item resulting in the required compliance of that object or data item. Modifications in Natural Engineer are classified according to Category and Type.

Refactoring

Improving a computer program by reorganizing its internal structure without altering its external behavior.

Soft Link

A Soft Link is where a link between two objects has been defined using an alphanumeric variable rather than a literal constant.

TLM

Text Logic Members are used to contain the code required to support inclusion of common code into the application. An example of this is the code to include into an application before updating a database.

Type

The Type of Modification available, for example: Data Item, Keyword and Literal.

Variable

A collective term used for any data fields within a programming object. These can be user-defined variables, DDM fields or System Variables. It is inter-changeable with the term ‘data item’.

Related Literature

The complete set of Natural Engineer manuals consists of:

1 **Natural Engineer Concepts and Facilities (NEE91-006ALL)**

The Concepts and Facilities manual describes the many application systems problems and solutions offered by Natural Engineer, providing some guidelines and usage that can be applied to Natural applications.

2 **Natural Engineer Release Notes (NEE91-008ALL)**

The Release Notes describe all the information relating to the new features, upgrades to existing functions and documentation updates that have been applied to Natural Engineer.

3 **Natural Engineer Installation Guide for Windows (NEE91-010WIN)**
Natural Engineer Installation Guide for Mainframes(NEE91-010MFR)
Natural Engineer Installation Guide for Unix (NEE91-010UNX)

The Installation Guide provides information on how to install Natural Engineer on PC, Unix and mainframe platforms.

4 **Natural Engineer Administration Guide (NEE91-040WIN)**
Natural Engineer Administration Guide (NEE91-040MFR)
Natural Engineer Administration Guide (NEE91-040UNX)

The Administration Guide provides information on all the various control settings available to control the usage of the different functions within Natural Engineer.

5 **Natural Engineer Application Management (NEE91-020WIN)**
Natural Engineer Application Management (NEE91-020MFR)
Natural Engineer Application Management (NEE91-020UNX)

The Application Management manual describes all the functions required to add Natural applications into the Repository.

6 **Natural Engineer Application Documentation (NEE91-022WIN)**
Natural Engineer Application Documentation (NEE91-022MFR)
Natural Engineer Application Documentation (NEE91-022UNX)

The Application Documentation manual describes all the available functions to document a Natural application within the Repository. These functions will help enhance / supplement any existing systems documentation such as BSD / CSD / Specifications etc.

7 **Natural Engineer Application Analysis and Modification (NEE91-023WIN)**
Natural Engineer Application Analysis and Modification (NEE91-023MFR)
Natural Engineer Application Analysis and Modification (NEE91-023UNX)

The Application Analysis and Modification manual describes all the available functions to carry out analysis of Natural applications; including basic keyword searches. The modification process is described and detailed to show how it can be applied to modify single selected objects within a Natural application, or the entire Natural application in one single execution.

8 **Natural Engineer Application Restructuring (NEE91-024WIN)**
Natural Engineer Application Restructuring (NEE91-024MFR)
Natural Engineer Application Restructuring (NEE91-024UNX)

The Application Restructuring manual describes the analysis and modification functionality required to carryout some of the more sophisticated functions such as Object Builder.

9 **Natural Engineer Utilities (NEE91-080WIN)**
Natural Engineer Utilities (NEE91-080MFR)
Natural Engineer Utilities (NEE91-080UNX)

The Utilities manual describes all the available utilities found within Natural Engineer and, when and how they should be used.

10 **Natural Engineer Reporting (NEE91-025ALL)**

The Reporting manual describes each of the reports available in detail, providing report layouts, how to trigger the report and when the report data becomes available. The various report-producing mediums within Natural Engineer are also described.

11 **Natural Engineer Batch Processing [Mainframes] (NEE91-026MFR)**
Natural Engineer Batch Processing [Unix] (NEE91-026UNX)

The Batch Processing manual describes the various batch jobs (JCL/Scripts) and their functionality.

12 **Natural Engineer Messages and Codes (NEE91-060ALL)**

The Messages and Codes manual describes the various messages and codes produced by Natural Engineer.

13 **Natural Engineer Web Interface Installation and Configuration Guide(NEA84-010ALL)**

The Web Interface Installation and Configuration Guide provides information on how to install and configure the Natural Engineer Web Interface.

14 **Natural Engineer Advanced Services (NEE91-017WIN)**
Natural Engineer Advanced Services (NEE91-017MFR)
Natural Engineer Advanced Services (NEE91-017UNX)

The Advanced Services manual describes various advanced options such as the Refactoring of Natural application source code with Natural Engineer, conversion of applications for Natural for Ajax, Business Rule processing and Data Masking.

1

HOW NATURAL ENGINEER WORKS

Chapter Overview

This chapter provides an overview of the processes that provide Natural Engineer functionality.

The topics covered are:

1. The Natural Engineer Package
2. Environment Process
3. Analysis Process
4. Modification Process
5. Utilities
6. Error Handling
7. Consistency Analysis
8. JCL Flow Process

The Natural Engineer Package

Natural Engineer consists of a number of processes, which, when executed sequentially, perform the Environment, Analysis and Modification functions.

- PC versions of Natural Engineer provide a PC-based Repository and a GUI front-end that allow you to specify the various required parameters and triggers for the individual processes.
- Mainframe versions of Natural Engineer provide a mainframe based Repository and mainframe screens that allow you to specify the various required parameters and triggers for the individual processes.

By default all users have access to all functions within Natural Engineer. If sites wish to restrict what functions users can access then they can set up User Profiles to limit functionality. User Profiles are available from the Administration menu on the PC and are controlled by settings in Global Properties.

Inventory

All Natural Engineer processing options start with the creation of the Natural Engineer Repository. By this process of extracting and loading application information into the database, Natural Engineer provides a comprehensive inventory of statistics, metrics, structures, cross-reference, high-level and detail-level data. In addition, Natural Engineer gives you the ability to review this inventory, interactively, or through reporting.

The creation of an Inventory requires the following processes:

- Extract Source Code
- Load extracted Source Code into the Repository

Natural Construct

Natural Engineer ‘understands’ and processes Natural Construct code.

Construct Models and User Exits are identified and lines of code are marked as either generated or User Exit code.

During each Natural Engineer process, Construct-specific reports are provided to show the effect of the process on the Construct Model, as well as to provide referencing information.

During the Modification phase, only those lines of code that are in User Exit code are modified. Generated Code is not modified.

If a customer has modified their Construct-generated object by removing the **SAG lines at the top of the program (thus making the object non re-generatable), Natural Engineer will treat the object as a normal non-generated object.

Predict Case

Natural Engineer ‘understands’ and processes Predict Case code.

Predict Case Components are identified and lines of code are marked depending on which Component they are part of.

An impact report is available which identifies which impacted lines of code are related to which Predict Case Component.

Steplib Processing

Natural Engineer can be used with applications that make use of steplib libraries during run time, where the steplib library may contain additional / common processes used by the application.

Natural Engineer needs to know the relationship between the application and the referenced steplib library, to maintain integrity during the Load, Impact and Modification processes.

Up to eight steplib libraries can be catered for per application.

The following example illustrates a simple steplib application process, based on the application library NATLIB1, which utilizes a steplib library NATSTEP1.

1. Define an application for the application library NATLIB1.
2. Specify the steplib relationship for application NATLIB1, by adding the steplib library name NATSTEP1 to the steplibs list using Application Properties.

Note: If the steplib library has not previously been loaded into the Repository, a message will be displayed when the Application Properties are being specified. This is for information purposes only. The steplib library only needs to be loaded if you wish to have a more complete picture of your Natural Environment.

3. Execute Extract Source Code and review the extract by checking the Quality Logs: Extract Source Code Summary and Missing Natural Objects.
4. Load the application NATLIB1 into the Repository. This will generate the cross-reference records for each object being referenced from the steplib library NATSTEP1, by objects within the application library NATLIB1. This data will be held as part of the application NATLIB1.

   There will also be cross-reference records created for each object on the steplib library NATSTEP1, which are referenced by objects in application library NATLIB1. This data will be held as part of the steplib application NATSTEP1.

   This provides a complete picture of the relationship between the application library NATLIB1 and steplib library NATSTEP1.

If you wish to load the steplib library, use the following steps:

1. Define an application for the steplib library NATSTEP1.
2. Execute Extract Source Code and review the extract by checking the Quality Logs: Extract Source Code Summary and Missing Natural Objects.
3. Load the application NATSTEP1 into the Repository.

Environment Process

This process creates the Natural Engineer Repository with the source code of the application to be analyzed. The scope of this process ensures that the most complete Repository information is generated on which to perform Analysis.

- The source code is extracted from the application for each statement, checked, neutralized, and loaded into the Natural Engineer Repository. This allows Natural Engineer to handle all versions of Natural application code.
- From the cross-referenced Repository created, Natural Engineer provides comprehensive application Analysis, with reports at summary, object and detail levels. These application Analysis reports provide inventory information about all the application code in the Repository. These reports can also be represented graphically using the interface to an OLE-compliant diagramming tool.
- The Environment process also performs a quality check on the code, reporting any missing objects or incomplete syntax.

In order to ensure that DDM field formats are available to Natural Engineer, it is recommended that you include all DDMs referenced by the application. After extraction of the source code, the Missing Natural Objects report also reports on missing DDMs. Before you continue, you must ensure that all relevant DDMs are extracted. Any objects that reference those DDMs must be re-extracted to include the DDM information.

Fundamental Steps

1. Create an application name and specify the application properties to be applied by identifying the Natural library from which the source code is to be extracted.
2. Define Extract Selection Criteria to extract either the complete library or selected objects.
3. Execute Extract, which will create an output file.
4. Load the Repository with the extracted code, which will load the file created by the Extract process.
5. Review the Extract Quality Logs to identify and correct errors.

   Note: For more information on the errors, refer to the section Error Handling.

6. Where errors have been identified, for errors prefixed with APP, identify the location of any missing objects and copy them to the application library, or a defined Natural Engineer steplib. These will be missing DDMs, Data Areas, or COPYCODE. Missing DDMs on the mainframe, mean they do not exist in Predict. Missing DDMs on the PC, mean they do not exist in the application library, a steplib library or the System library.

   Investigate source code syntax errors identified for correction or removal of the object.

   If a Natural Engineer Extract does not end cleanly because of logic errors or invalid response codes from Natural or Adabas it will issue a return code of 255. If the extract identifies missing objects or a non-critical error only then a return code of 254 is returned.

7. Where missing objects are required for Natural Engineer processing, either selectively Extract and Load the relevant missing objects identified from the Extract Quality Log (that is, execute steps 2 to 4 again for these objects), or execute Extract Missing Object once the objects have been copied.

8. Repeat the above steps until all required objects are no longer identified as errors on the Extract Quality Log.

9. Review the Load Quality Log for any errors that occurred during loading the Natural Engineer Repository. Errors in this log can also relate to the database session parameters on which the Natural Engineer Repository file resides.

   Note: For more information on the errors, refer to the section Error Handling).

10. Review the Application Reports as required you can choose between summary, object and detail reports.

Analysis Process

The Analysis process provides the ability to define a set of Search Criteria for scanning and identifying impacts in the Application, for statements, keywords, fields and text, in order to find code that may need to be changed for maintenance, standards, migrations or componentization.

Known Search Criteria may be provided, for example the Analysis and changes required for migrating versions of Natural. Where Consistency is specified for Analysis, for each field identified as a result of the process, Natural Engineer performs unlimited Iterations (or limited to the number specified in the NATENG.INI/CINI settings), using left and right argument checking until all possible combinations are exhausted.

The Impact Analysis provides the information to ascertain the size of the impact for the application, as well as the level of automatic Modification available. Reports of the Impact Analysis are generated at summary, object and detail levels, and impacted source code is highlighted.

Note: For more information on NATENG.INI settings refer to Chapter 1 in the Natural Engineer Administration Guide for Windows manual.

Note: For more information on CINI settings refer to Chapter 1 in the Natural Engineer Administration Guide for Mainframes manual

Fundamental Steps

1. Define the Impact Search Criteria and any applicable Modification strategy, and specify whether a Consistency check is required. The Impact Search Criteria can be versioned so that multiple sets of criteria are available per application loaded into the Repository.
2. Execute Impact Analysis.
3. Review the Data Item Impact Inventory report.
4. In order to review the Impacts identified, you can review the code on the Impact Element Maintenance screen, or view the highlighted Impacted Source Code in Browser by selecting View Impacted Source Code.
5. Execute the Impacted External Objects report to determine if any impacted data elements are passed to external routines. The owner of the external routine must be identified to determine what effect the impacted data element has on the external object and how it may need to be changed.
6. Review Impact Reports as required: you can choose between summary, object and detail reports.

Modification Process

The Modification process allows you to confirm and modify the categories of Modification applied by Natural Engineer. Categories of Modification are: Automatic, Manual, Reject, Not Applicable and Generated.

Before executing the Modification, you can view and confirm the code and definition changes to be applied. Reports of Modification changes are generated at the summary, object and detail levels, as well as change audit logs.

Fundamental Steps

Note: After Modification, Natural Engineer places all modified code into the Modification Library as specified in the Application Properties for the application. If this is not set, Natural Engineer places all modified code in a library name with an 'X' as the last character of the application name. If the name is already 8 characters long, the last character is removed and replaced with the 'X'.

Warning: Once you have started modifying objects with Natural Engineer, do not re-execute the Impact Analysis until the Modification phase has been completed. Current Modification information within Natural Engineer is over-written if Impact Analysis is re-executed.

1. Review Modification categories and types by using the , Modification Element Maintenance option.
2. Verify and modify the default category and type changes for each object as required. Assign the Reject category to any change not required. Natural Engineer does not check parents of fields when changes are made to the Category and Type, so these must be made manually.
3. Execute automatic modification either by object using the Modification Element Maintenance screen, or for all objects using the Execute Modification for All Objects function on the menu.
4. Execute the Impacted Objects Not Directly Modified report to determine impacted objects that contain no direct Modification lines that must be copied to the library containing the modified code. These are Objects that reference impacted data areas or Copycode.

5. Execute the Predict Changes report and make changes to Predict definitions so that the new DDMs can be generated. Allow access to the new DDMs from the library containing the modified code.
6. Stow the Data Areas.
7. Execute the ‘check’ command in Natural to determine if any errors exist in the code which may require manual change, such as overlapping fields in maps. This verifies the automatic Natural Engineer Modification.
8. Make any further manual changes required to the application objects identified by Natural Engineer.
9. ‘Stow’ the application code in the modified library.
10. Review Modification Reports for information of interest, including summary, object, and detail reports.
11. Audit information is available by using the Change Management Tracking option from the Utilities menu.

Utilities

Natural Engineer provides several Utility options that complement the core processes:

- **Task Scheduler**
  Provides the facility to schedule unattended tasks for long running Environment, Analysis and Modification batch tasks. Long running tasks can be scheduled to run out of normal working hours to provide less disruption to the normal working day.

- **Compare**
  Provides the facility to compare Natural objects between two or three Natural libraries. The Compare Results show differences between objects using customizable color schemes. Code differences can be copied into the base object using basic editing functions, copy and paste. User changes can be applied using basic editing functions insert, delete and undo delete. Changes can then be saved to a Natural library.

  Note: This option is only available to the Windows version of Natural Engineer.

- **Mode Conversion**
  Provides the facility to convert Natural Reporting mode objects into Natural Structured mode objects.

- **Change Management Tracking**
  Provides Audit Trail information on source code changes applied to objects using Natural Engineer's Modification option. Information is available either in interactive mode or via reports using display medium of Microsoft Word, a spreadsheet e.g., Microsoft Excel, Adobe PDF, HTML or the Natural screen.

- **Keyword Catalogue**
  Provides the ability to identify related items within the Natural Engineer Repository by defining keywords.

- **Architectural Governance**
  Provides the ability to apply global or application specific coding standards to Natural Objects.

Error Handling

Environment

Natural Engineer may generate two error logs during execution, one when extracting source code and one when loading the Natural Engineer Repository.

The error logs have the following prefixes:

- **APP**
  This error type identifies:
  - missing Natural objects within the application library. They could be DDMs, Data Areas or Copycode. You can resolve these errors by moving the identified object to the application library, a defined steplib library or the SYSTEM library. These can then either be selectively Extracted and Loaded, or you can execute Extract Missing Objects, or re-Extract and Load the application.
  - basic application syntax errors, which are further identified by the error text: SRC ERR. Investigate source code syntax errors and correct, or remove the object.
- **GSL**
  This error type relates to the actual processing of the application code. The error relates to both a Natural Engineer object as well as an object in the user application.

Support

Before you contact the Support Desk on any error log, please take the following actions:

1. Investigate whether the user application object can be ‘checked’ within the Natural environment.
2. If the check fails, then either make appropriate changes to the object, or remove it from the library. No further action is required.
3. If the object ‘checks’ under Natural, then provide a copy of the relevant line of code with the appropriate error log to the Support Desk.

Consistency Analysis

Consistency Analysis is available when the consistency option is switched on when specifying Impact Search Criteria for search keywords DATAITEM, DBFILE and DEFINITION, or when using the search keyword MULTI SEARCH.

Consistency Analysis provides the relationship information for a specified Impact Search Criteria in relation to other data items within an impacted object. It will also analyze across objects to build up a full relationship of the specified criteria across all impacted data items, across all objects within an application.

There are three distinct phases during this type of Analysis:

1. Single Object Impacts
2. Inter Object Relationships (IOR)
3. Propagation Phase

Single Object Impacts

The Impacted Object phase will search for the specified Impact Search Criteria and if a match is found within an object, the Impact is marked as ‘specified’. Then the Impact process will analyze the impacted object for any data manipulations of the specified criteria.

If the impacted data item is moved to another data item, then this target data item is impacted as ‘derived’. The derivation occurs on the second-n Iterations until all possible derivations have been exhausted. For example:

For the statement: **MOVE #A TO #B**

Using Impact Search Criteria: Search Keyword = **DATAITEM** and Search Value = **#A**, Impact Analysis would Impact **#A** as the specified Impact and **#B** as the derived Impact.

If **#B** was then further manipulated by statement: **MOVE #B TO #C,** then **#C** would be impacted as derived.

Inter Object Relationships (IOR)

Once the single object impacts have been completed the IOR processing is activated which checks for impacts across object boundaries. This part of Consistency Analysis is controlled via the IOR parameter setting in the NATENG.INI file. IOR can be turned off by setting IOR=N.

Given a Search Keyword of DATAITEM and a search value of ?#DATE? and using the following program and subprogram below

```
PGM1                                     SUBPGM1
DEFINE DATA LOCAL                        DEFINE DATA PARAMETER
01 #DATE(N06)                            01 #A(N06)
END-DEFINE                               LOCAL
CALLNAT 'SUBPGM1' #DATE                  01 #B(N06)
END                                      END-DEFINE
                                         MOVE #A TO #B
                                         END
```

Single Object Impact will only have marked #DATE as being impacted. IOR knows that #DATE is passed to SUBPGM1 and that the receiving field is #A. It checks to see if #A is already impacted, if not it impacts it as a derived impact.

Natural Engineer then has to check all derivations of #A and as such will also mark #B as derived in SUBPGM1. It will perform this IOR checking for all fields in the following statements:

- CALLNAT
- PERFORM (External subroutines only)
- STACK
- FETCH/FETCH RETURN
- RUN/RUN REPEAT
- INPUT USING MAP/FORM
- WRITE USING MAP
- INPUT USING HELP
- OPEN DIALOG

It will also try and match impacts 'from target to parameter' so for example if the receiving field in the subprogram is impacted but the parameter being passed is not, then Natural Engineer will mark the parameter field as being impacted.

Propagation Phase

The Propagation phase comprises six stages:

1. **Propagate Data Areas** where any impacts in code are propagated back to external data areas if they didn't previously exist.
2. **Propagate DDMs** where any impacts found within views within the code are propagated back to external DDMs.
3. **Propagate Copycode** where any impacts in code are propagated back to external Copycodes if they didn't previously exist.
4. **Propagate Definitions** where impacts to locally defined view fields are pushed back to external DDMs.
5. **Global Data Areas** where impacted Global data item(s) have been impacted, each Global data item will then be analyzed using the three phases. This process is optional and is controlled by the GLOBAL_DATAITEM= parameter setting in the NATENG.INI file.
6. **DDMs** where impacted DDM data item(s) have been impacted, each DDM data item will then be analyzed using the three phases. This process is optional and is controlled by the DDM_DATAITEM= parameter setting in the NATENG.INI file.

Special Features when using Consistency Analysis

Mapping of Redefined Fields

If an impacted field is a child of a redefined field, then Natural Engineer assumes that the bytes that the field occupies are impacted and will mark other fields that occupy these bytes as impacted as well.

For example:

```
DEFINE DATA LOCAL
01 #WORK-AREA(A20)
01 REDEFINE #WORK-AREA
    02 #FIELD-1(A10)
    02 #FIELD-2(A06)
    02 #FIELD-3(A04)
01 REDEFINE #WORK-AREA
    02 #FIELD-4(A10)
    02 #DATE-1(A06)
    02 #FIELD-5(A04)
END-DEFINE
WRITE #DATE-1
END
```

Impact Search Criteria: Search Keyword = **DATAITEM**, Search Value = **?DATE?** and consistency switched on.

Natural Engineer will mark **#DATE-1** as a specified impact. **#DATE-1** occupies bytes 11-16 inclusive of the field **#WORK-AREA**. Natural Engineer will then map this impact over all redefinitions of **#WORK-AREA**. The field **#FIELD-2** occupies the same byte range of **#WORK-AREA** so Natural Engineer will mark this field as a derived impact.

This facility may be turned off by the REDEFMAP= parameter setting in the [IMPACT] section of the NATENG.INI file. The default setting is REDEFMAP=N.

Deriving Parents of Redefined Fields

If a field is a child of a redefined field and is impacted Natural Engineer will derive the parent field.

For example:

```
DEFINE DATA LOCAL
01 #WORK-AREA(A16)
01 REDEFINE #WORK-AREA
    02 #FIELD-1(N08)
    02 #FIELD-2(N06)
    02 REDEFINE #FIELD-2
      03 #YY   (N02)
      03 #MM   (N02)
      03 #DD   (N02)
    02 #FIELD-3(N02)
END-DEFINE
WRITE #YY
END
```

Impact Search Criteria: Search Keyword = **DATAITEM**, Search Value = **?YY?** and consistency switched on.

Natural Engineer will mark the field **#YY** as being impacted and then derive the field **#FIELD-2**.

COMPRESS and SEPARATE Statements

If a source field is impacted in a COMPRESS or SEPARATE statement then Natural Engineer will only derive the target field. It will NOT derive all other source fields. However, if the target field is impacted Natural Engineer will not derive any source fields.

Source Field impacted

For example, using the following lines of code and a Search Keyword of **DATAITEM** and a Search Value of **?DATE?** and consistency switched on.

```
COMPRESS 'TODAY IS:' #DATE ':' #END INTO #ALPHA LEAVING NO
```

For this COMPRESS statement Natural Engineer will impact **#DATE** as a specified impact and it will then derive the target field **#ALPHA**.

```
SEPARATE #X INTO #5 #DATE1 #6 WITH DELIMITER ':'
```

For this SEPARATE statement Natural Engineer will impact **#DATE1** as a specified impact and it will then derive the target field **#X**.

Target Field impacted

However if a target field is impacted then Natural Engineer will not derive any source fields.

For example, using the following lines of code and a Search Keyword of **DATAITEM** and a Search Value of **?DATE?** and with consistency switched on.

```
0330 COMPRESS #7 #8 #9 INTO #DATE-ALPHA
```

For this COMPRESS statement Natural Engineer will impact **#DATE-ALPHA** as a specified impact. No other fields will be impacted.

```
SEPARATE #DATE1 INTO #1 #2 #3 WITH DELIMITER ':'
```

For this SEPARATE statement Natural Engineer will impact **#DATE1** as a specified impact and it will then derive the source fields **#1**, **#2** and **#3**.

JCL Flow Process

JCL Flow allows the display of forward or backward tracing of JCL from information contained within third-party scheduling tools such as CA-7 or OPC.

To set up JCL Flow Chain information the following steps need to be addressed:

1. Set up Trigger Types

   Triggers may be described as any action that causes a Job to be executed. For example a site may have a piece of standard JCL that may execute other jobs or use a scheduling tool to perform certain actions. Each of these individual actions could be described as a trigger. Trigger Types are set up on the JCL Triggers tab of the Global Properties screen.

   For further information on defining trigger types please see the Global Properties section of the Natural Engineer Administration Guide.

2. Delete any existing JCL Flow metadata by running stand-alone program JCFLDL-P from library SYSNEE. This program accepts one parameter of the Natural Engineer Application Name.

3. Create an input file containing JCL Flow meta-data.

   This would be created by the third-party scheduling tool e.g., from CA-7 the following command could be issued

   LJOB,LIST=NODD,JOB=*xx**

   Where ‘*xx’* is the prefix of the Job Names as specified within CA-7 that you wish to load into Natural Engineer.

   This will create a file which will used to add the JCL Flow detail into the repository via the NEEAPI1 API.

4. Create a program to parse the output from the scheduler and call the JCL Flow API ,NEEAPI1N, to load the data into the repository. A sample program for CA-7, NEEAPI1P, is provided in library SYSNEE. This should be configured to match the user’s requirements.

   Information that you may want loaded into Natural Engineer include:

   - Last Run Date/Time
   - CPUTM – Average CPU time
   - ELAPTM – Average Elapsed time
   - Scheduled ID
   - System Name (NDM,SECS,INFOPAC, etc)
   - Active/Inactive

5. The information can then be viewed via the JCL Flow Chain diagrams.

   For further information on JCL Flow Chain diagrams please see the Natural Engineer Management Guide.

2

COMMON FACILITIES

Chapter Overview

This chapter describes some of the common facilities available within Natural Engineer.

The topics covered are:

1. General Selection
2. Remote Development Environments

General Selection

The General Selection is a multi-purpose screen, which provides a mechanism to assist in selection decisions within various Natural Engineer options.

The basic functionality of the General Selection screen is to provide a list of items relating to the option that has invoked the screen.

The types of lists available include:

- Natural Libraries
- Natural Objects
- Data Items
- Data Definition Modules (DDMs)
- Data Definition Module Fields
- Text Logic Members (TLM)
- PAC Applications
- PAC Statuses
- Cobol Libraries
- JCL Libraries
- CICS Objects

Note: The PAC Applications and Statuses options are only available if Natural Engineer is executing in a remote development environment and PAC version 2.4.2 or above is installed on the mainframe.

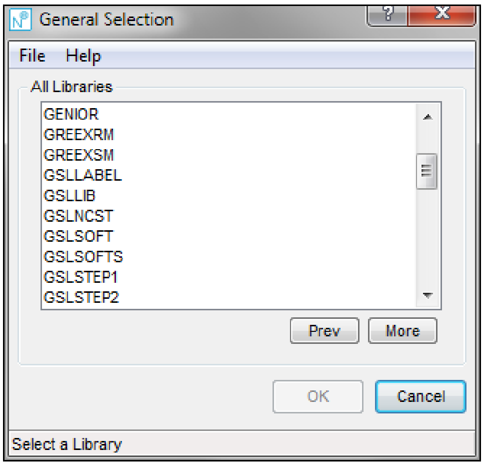
General Selection Window

The General Selection screen utilizes a common format used for all types of selection lists. The filtering options are only displayed where applicable and are tailored to the particular type of selection and the contents of the repository.

A Context menu is available to reposition the item list by using the right hand mouse button with a single click.

Samples of the General Selection screen are now illustrated. The subsequent description is only shown once, but equally applies to all variations.

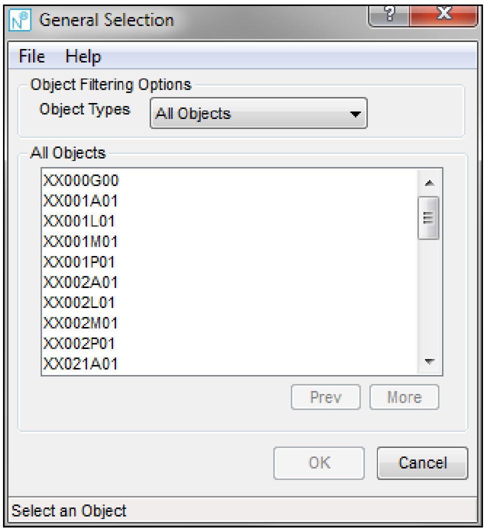
The following Figure 2-1 illustrates the General Selection screen displaying Natural Libraries.



Select a Library

Figure 2-1 General Selection screen displaying Natural Libraries

The following Figure 2-2 illustrates the General Selection screen displaying Natural Objects.



Object Filtering Options

Object Types: All Objects

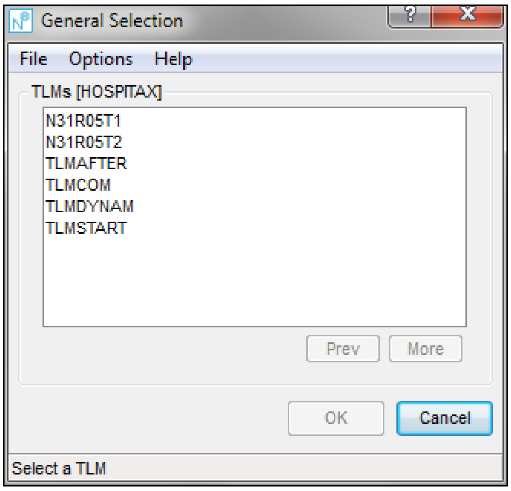
All Objects

XX000G00
XX001A01
XX001L01
XX001M01
XX001P01
XX002A01
XX002L01
XX002M01
XX002P01
XX021A01

Select an Object

Figure 2-2 Natural Engineer Selection screen displaying Natural Objects

The following Figure 2-3 illustrates the General Selection screen displaying TLMs.



TLMs [HOSPITAX]

N31R05T1
N31R05T2
TLMAFTER
TLMCOM
TLMDYNAM
TLMSTART

Select a TLM

Figure 2-3 General Selection screen displaying TLMs

MENU ITEMS	OPTIONS	DESCRIPTION
File	**Exit**	Exit the General Selection screen and return back to the previous screen.
Options	**Show TLMs from SYSTEM**	Display TLMs from the Natural library SYSTEM.
	Show TLMs from Modification Library	Display TLMs from the Modification Library as defined in the Application Properties screen.
	Note: The Options menu is only available when displaying TLMs.	
Help	Invoke the General Selection help.	

BUTTON NAME	DESCRIPTION
Prev	Scrolls the item list to previous page. This button will be available/unavailable depending on the value specified in the LISTBOXMAX parameter in the NATENG.INI file.
More	Scrolls the item list forward one page. This button will be available/unavailable depending on the value specified in the LISTBOXMAX parameter in the NATENG.INI file.

STATUS BAR ITEM	DESCRIPTION
Pane	Any General Selection processing messages.

SCREEN ITEMS	DESCRIPTION
Items List	List of items. The items can be any of the following: ▪ **Natural Libraries** ▪ **Natural Objects** ▪ **Data items** ▪ **Data Definition Modules** ▪ **Data Definition Module Fields** ▪ **Text Logic Members** ▪ **Cobol Libraries** ▪ **JCL Libraries** ▪ **PAC Applications** ▪ **PAC Statuses** ▪ **CICS Objects** ▪ **CICS Transactions** Items can be selected by double clicking with the left hand mouse button, or by a single click of the left hand mouse button and then using the '**OK**' button. *Note: For further information, refer to the section Invoking the General Selection Window*.
Object Filtering group:	
Object Types	Allows you to select the types of object to be listed. The object types displayed are tailored to the type of selection requested and the contents of the repository. *Note: The Object Types menu is only available when displaying Objects.*
Language	Allows you to select the programming language of the objects to be listed. The language types displayed are tailored to the types of objects present. *Note: The Language menu is only available when displaying Objects*.
Field Types	Allows you to select the types of fields to be listed. Available selections are: ▪ **All Fields** ▪ **Non-Database Fields** ▪ **Database Fields** ▪ **System Variables** *Note: The Field Types menu is only available when displaying data items.*

<table>
<tr><th>CONTEXT MENU ITEM</th><th>DESCRIPTION</th></tr>
<tr><td><b>Change Start Position of XXX List…</b></td><td>
<p>Reposition the list of 'XXX' items to start from a particular name, where 'XXX' will contain the type of list being displayed in the General Selection screen.</p>
<p>The reposition value can be input using either a complete name or part name using an '*' (asterisk) wildcard.</p>
<p>The reposition value is appended to the object list title to highlight the type of repositioning being applied.</p>
<p>Possible reposition values are:</p>
<table>
<tr><th>Value</th><th>Result</th></tr>
<tr><td><b>' ' (blank)</b></td><td>Reposition to the top of the object list.</td></tr>
<tr><td><b>*</b></td><td>Reposition to the top of the object list.</td></tr>
<tr><td><b>ABC*</b></td><td>Only show objects that are prefixed by 'ABC'.</td></tr>
<tr><td><b>XYZ</b></td><td>Reposition to the first object that either matches or is greater than 'XYZ' and then continue the object list from that point.</td></tr>
<tr><td colspan="2">The following reposition values are for System Variables only:</td></tr>
<tr><td><b>**</b></td><td>Reposition to the top of system variable list.</td></tr>
<tr><td><b>*CURS</b></td><td>Reposition to the first system variable that either matches or is greater than '*CURS' and then continue the system variable list from that point.</td></tr>
<tr><td><b>*DAT*</b></td><td>Only show system variables that are prefixed by '*DAT'.</td></tr>
</table>
</td></tr>
</table>

Note: For more information on the NATENG.INI file parameter LISTBOXMAX refer to Chapter 1 in the Natural Engineer Administration Guide for Windows manual.

Invoking the General Selection Window

The General Selection screen can be invoked by using the selection button found adjacent to the input boxes, within applicable Natural Engineer screens.

The following Figure 2-8 illustrates the selection button used to invoke the General Selection screen.

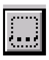


Figure 2-8 Selection button used to invoke the General Selection screen

The following Figure 2-9 illustrates the selection button as it appears on the Application Properties screen.

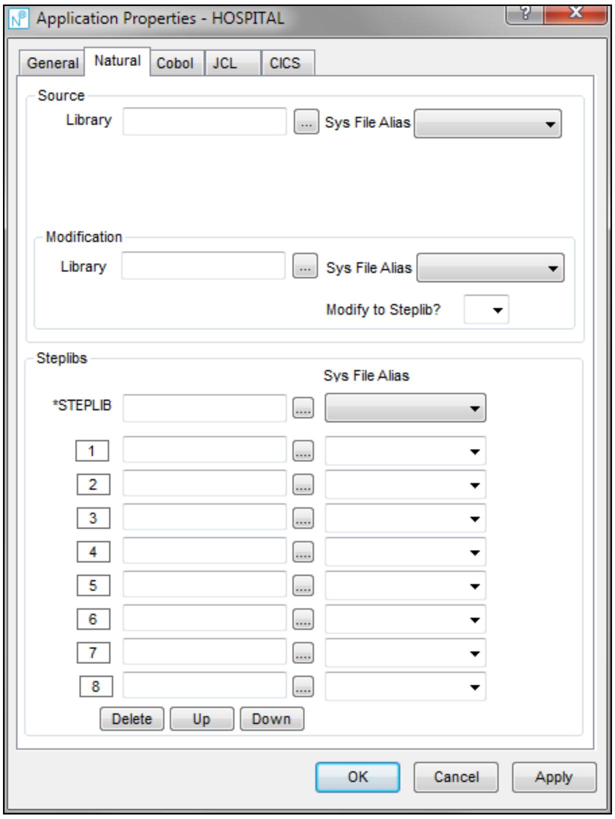


Figure 2-9 Selection button as it appears on the Application Properties screen

TLM List Context Menu

When using the General Selection screen to list TLMs, a context menu is available by using the right hand mouse button with a single click.

This option provides the facility to view the contents of the selected TLM using the GenSource source code window.

The following Figure 2-10 illustrates the TLM list context menu.

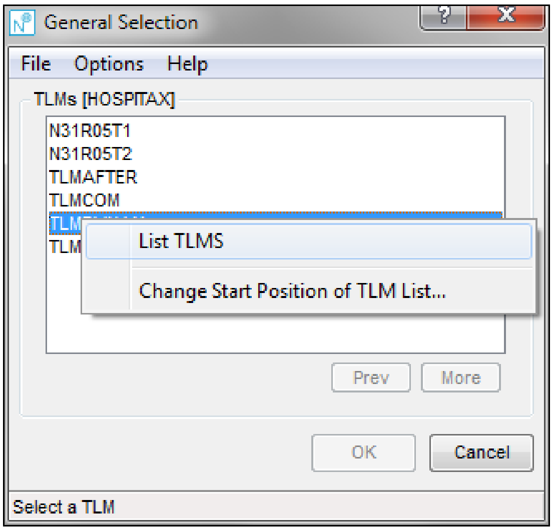


Figure 2-10 TLM list context menu

Remote Development Environments

Natural Engineer offers compatible support for remote development environments using Natural's Single Point of Development (SPoD) available with Natural version 8.3.

This facility provides the mechanism to access Natural source code libraries residing on remote mainframe or unix computers, whilst retaining the full functionality and GUI of Natural Engineer for Windows.

This in-turn provides full Natural Engineer graphical functionality previously only available in Natural Engineer for Windows, to the mainframe or unix environment.

Using the SPoD environment, Natural Engineer will read Natural source code from the remote environment. Any modifications applied using Natural Engineer's modification options will be written back to the relevant Natural libraries on the remote environment.

It also logically follows that if the SPoD environment is being used, then the Natural Engineer Repository file being used should reside on the mainframe or unix environment.

Note: When using Natural Engineer in a SPoD environment, Natural Engineer version 6.1 or above must be installed in the Windows environment being used at run time. The same version of Natural Engineer must also be installed in the mainframe environment being used at run time.

If the remote environment is unix then Natural Engineer 6.1.2 or above must be installed in the Windows environment being used at run time. The same version of Natural Engineer must also be installed in the unix environment being used at run time.

Invoking Natural Engineer under Natural Version 8.3

There are two ways to invoke Natural Engineer under Natural version 8.3:

1. **Local**

   Natural version 8.3 is started and the start up program "NEESTART" invoked from the Natural command line. This will invoke Natural Engineer using the local environment for the machine on which it is being executed. All Natural source code will be referenced from the local FUSER.

2. **Remote**

   Natural version 8.3 is started and the remote mainframe environment needs to be mapped (i.e., logged onto from within Natural version 8.3). The start up program "NEESTART" is then invoked from the Natural command line. This will invoke Natural Engineer using the remote environment that has been mapped. All Natural source code will be referenced from the remote environment FUSER.

   To enable the source code to be correctly displayed in this mode the mapping to the remote environment should have CFICU=ON,CP=AUTO as parameters.

To assist the User as to which mode (local or remote) is being utilized there is new status bar pane available on the main Natural Engineer screen. This will show 'Local' for the local FUSER environment, and 'Remote' for the remote environment FUSER.

Note: This will only appear if Natural version 8.3 has been installed.

Note: For more information on Natural's Single Point of Development for remote development on a mainframe computer refer to the SPoD specific documentation on the Natural documentation CD.

Batch Job Submission

Certain Natural Engineer functions require execution in a batch environment to perform their specific processing.

When running Natural Engineer in a SPoD environment it is possible to submit batch jobs to execute in the mapped mainframe, windows or unix environment from the GUI of Natural Engineer for Windows.

Certain Job Submission parameters e.g., Job Name, Job Class and SYSID(VSE/ESA only) may be overridden on the client platform if the “Allow Job Submission Overrides” flag is set on. This is set in the Global Properties screen of the Global Settings option from the Options/Administration menu of the main Natural Engineer window.

The progress of the submitted batch jobs can be monitored using the Job Processing Log window.

The batch process is controlled by the application control record, which is locked at the time of job submission, modified during the job execution as each job step completes and released at the completion of the job.

In the event of any serious failures, the application control record is not released resulting in no further batch job submission possible until the application is released.

The application can be released by using the option Release Application Lock from the Options menu, or, by selecting the required Natural Engineer batch function, when the Release Application Lock screen will be displayed.

The following table summarizes the Natural Engineer batch functions:

Menu	Function
Application	Delete
Environment	Extract Source Code
	Load Repository
	Extract & Load
	Extract, Load & Impact
	Extract Missing Objects
Analysis	Impact Execution
Modification	Execute Modification for All Objects

A batch job is submitted by selecting any of the eligible Natural Engineer batch functions.

Once the batch job is submitted, a confirmation message will appear in the Main Natural Engineer screen status bar message area confirming that the batch job has been submitted successfully.

The confirmation message will contain the job name that has been used, this will be the User-id used to sign on to the SPoD environment. This job name should be used to locate the job on the remote environment.

Job Processing Log Window

The Job Processing Log screen is displayed when the batch job is submitted for any of the Natural Engineer batch functions.

The Job Processing Log screen provides information on the progress of the batch job that has been submitted. The information is obtained from the application control record.

The title bar will reference the name of the Natural Engineer function that is executing.

Note: The Job Processing Log screen does not interact with the batch job directly. It is recommended that the submitted batch jobs are checked manually for any job outputs and/or failures.

MENU ITEMS	OPTIONS	DESCRIPTION
File	**Always on top**	This option allows you to control the display position of the Job Processing Log screen. If selected (indicated by a tick to the left of the text) it will always keep the Job Processing Log screen in the foreground. If de-selected (no tick) then the Job Processing Log screen can be moved to the background.
	Save Job Processing Log…	The Job Processing Log details can be saved as a PC text file. The default name is ‘JOBLOG.TXT’. By default, this file will be saved to the data folder where Natural Engineer is installed.
	Exit	Exit the Job Processing Log screen.
Help	**About NEEJobLog**	Display the NEEJobLog version information.

SCREEN ITEMS	DESCRIPTION
Job Processing Log Entries	The Job Processing Log entries show the progress of the submitted batch job. They are automatically refreshed and are for information purposes only. The details are obtained from the application control record, which is updated during the batch job execution. The entries are listed in descending date/time order. When the batch job has completed, the final message will read: “Job Finished” *Note: The Job Processing Log screen does not interact with the batch job directly. It is recommended that the submitted batch jobs are checked manually for any job outputs and/or failures.*

BUTTON NAME	DESCRIPTION
Cancel	Cancels the Job Processing Log process and closes the Job Processing Log screen. *Note: This does NOT cancel the batch job.*

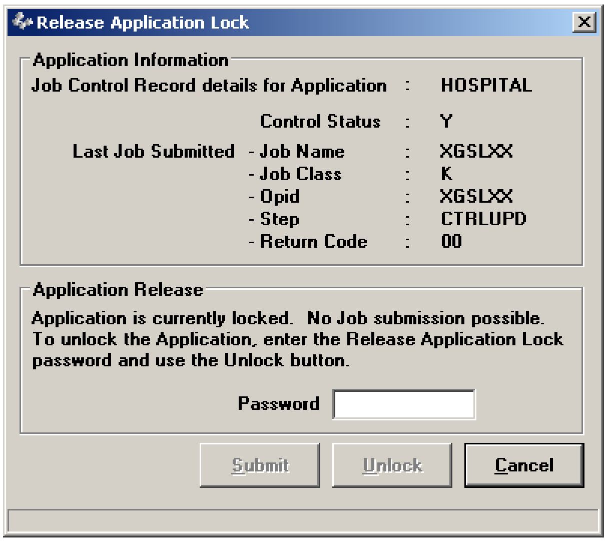
Release Application Lock

If an application has had a serious failure during the batch job execution, then the application control record will be locked preventing any further batch job submission for that application.

Before any further batch jobs can be submitted for the application, the application needs to be unlocked using the Release Application Lock option.

This can be invoked either by selecting menu Options➔Release Application Lock, or, by selecting any of the Natural Engineer batch functions. This second option provides the facility to submit a job from the Release Application Lock screen.

The following Figure 2-16 illustrates the Release Application Lock screen.



Job Control Record details for Application : HOSPITAL

Control Status : Y

Last Job Submitted - Job Name : XGSLXX
- Job Class : K
- Opid : XGSLXX
- Step : CTRLUPD
- Return Code : 00

Application is currently locked. No Job submission possible.
To unlock the Application, enter the Release Application Lock password and use the Unlock button.

Password

Figure 2-16 Release Application Lock screen

SCREEN ITEMS	DESCRIPTION
Application Information group:	
Application	The name of the application being processed.
Control Status	This will be set to ‘**Y**’ (locked). After releasing the application this will update to show ‘**N**’ (unlocked).
Job Name	The Job Name used for the last submitted job.
Job Class	The Job Class used for the last submitted job.
Opid	The opid used for the last submitted job.
Step	The last completed job step.
Return Code	Always set to ‘**00**’.
Application Release group:	
Password	The password is set to '**GENRJE01**' by default. The password may be changed by modifying the Natural Engineer User Exit, NEEUEX1 which is located in the SYSNEE library.

BUTTON NAME	DESCRIPTION
Submit	Submit the batch job. *Note: This button is only enabled after the application has been unlocked. If the Release Application Lock option has been selected from the Options menu, then this button will not be enabled.*
Unlock	Releases the locked application. The control status will be updated to ‘**N**’ (unlocked). Batch jobs can now be submitted again for the application. *Note: This button is only enabled when the correct password is input.*
Cancel	Cancel the Release Application Lock process and return back to the main Natural Engineer screen.

3

USER EXITS & APPLICATION PROGRAMMING INTERFACES

Chapter Overview

This chapter describes the user exits and application programming interfaces (API's) available within Natural Engineer.

The topics covered are:

1. User Exits
   a. NEEUEX1: Application Lock Release Password
   b. NEEUEX2: RJE Online Job Submission
   c. NEEUEX3: Modification
   d. NEEUEX4: Generate Adapter Name for NJX Conversion
   e. NEEUEX5: Generate Application Names
   f. NEEUEX6: Bespoke Security Interface
2. Application Programming Interfaces
   a. NEEAPI1N: JCL Flow
   b. NEEAPI2N: Extract and Load Services Data
   c. NEEAPI3N: Add Links to Services

User Exits

Natural Engineer provides various User Exits to allow the user to customize their Natural Engineer installation.

The user exit modules are supplied named ‘NEEUEXnX’ on the Natural Engineer SYSNEE library. This is to avoid overwriting any existing (modified) versions on the production SYSNEE library during Natural Engineer installation. If this user exit has not been loaded before, then it will need to be renamed to ‘NEEUEXn’ before making use of the User Exit Modification functionality.

The User Exits available are:

1. NEEUEX1: Application Lock Release Password
2. NEEUEX2: RJE Online Job Submission
3. NEEUEX3: Modification
4. NEEUEX4: Generate Adapter Name for NJX Conversion
5. NEEUEX5: Generate Application Names
6. NEEUEX6: Bespoke Security Interface

NEEUEX1: Application Lock Release Password

A user exit is available to change the default Application Lock Release Password on mainframe and unix platforms.

The default value is 'GENRJE01'. Any local changes required can be applied directly to NEEUEX1 and then stowed on SYSNEE. The new value must be uppercase and contain no special characters.

NEEUEX2: RJE Online Job Submission

A user exit is available for use with the NATRJE function on mainframe and unix platforms, which allows any local standard security issues to be controlled.

The default code in the user exit will call NATRJE directly. Any local changes required can be applied directly to NEEUEX2 and then stowed on SYSNEE.

The following Figure 3-1 illustrates the default NEEUEX2 source code.

```
0010 DEFINE DATA PARAMETER
0020 01 #NATRJE-CARDS        (A80/1:50)
0030 01 #NATRJE-COUNT        (B4)
0040 01 #NATRJE-FLAG         (A1)
0050 01 #NATRJE-RETHEX       (B2)
0060 END-DEFINE
0070 /*
0080 CALL 'NATRJE' #NATRJE-CARDS(1:50)
0090   #NATRJE-COUNT
0100   #NATRJE-FLAG
0110   #NATRJE-RETHEX
0120 /*
0130 END
```

Figure 3-1 Default NEEUEX2 source code

NEEUEX3: Modification

The user exit module can be customized to replace selected items, which have been impacted by the Impact Analysis process, within an object with up to 20 lines of new source code. It is available on PC, mainframe and Unix platforms.

The user exit module can be invoked during the Modification process by selecting an impacted statement line in the Impact item list on the Modification Element Maintenance screen and changing the Modification Category to ‘X’ (User Exit).

Example of User Exit Modification

This example is based on the sample code from the supplied user exit module in library SYSNEE. The sample code illustrates how to change a CALLNAT ‘XXGETID’ statement to be CALLNAT ‘SAGGETID’.

Impact Analysis is run against the HOSPITAL application using the following Impact Criteria settings of DBFILE (Impact Type), PATIENT (Keyword Value) and PATIENT-ID (Search Value).

The impacted item at statement line number 0690 within object XX021P01 is selected and the Modification Category changed to ‘X’ (User Exit).

The Modification process is then invoked for object XX021P01.

The following Figure 3-2 illustrates the sample source code for object XX021P01 before Modification:

```
0630 *
0640 SET KEY ALL
0650 MOVE *DATN TO #L-TEMP-DATE
0660 DECIDE ON FIRST VALUE OF #G-SELECTED-OPTION
0670 VALUE "A"
0680   MOVE *DATX TO PATIENT.RELEASED
0690   CALLNAT "XXGETID" PATIENT.PATIENT-ID #L-TEMP-DATE-N6
0700   MOVE "              ADD A PATIENT" TO #M-MAP-HEADING
```

Figure 3-2 Sample source code for object XX021P01 before Modification.

The following Figure 3-3 illustrates the sample source code for object XX021P01 after Modification:

```
0630 *
0640 SET KEY ALL
0650 MOVE *DATN TO #L-TEMP-DATE
0660 DECIDE ON FIRST VALUE OF #G-SELECTED-OPTION
0670 VALUE "A"
0680   MOVE *DATX TO PATIENT.RELEASED
0690 /*   CALLNAT "XXGETID" PATIENT.PATIENT-ID #L-TEMP-DATE-N6 /* NEE OLD
0700 *                                                        /* Changed
0710 * Start of Block of Code from NEEUEX3                    /* Changed
0720 *                                                        /* Changed
0730   CALLNAT "SAGGETID" PATIENT.PATIENT-ID #L-TEMP-DATE-N6    /* Changed
0740 *                                                        /* Changed
0750 * End of Block of Code from NEEUEX3                      /* Changed
0760 *                                                        /* Changed
0770   MOVE "              ADD A PATIENT" TO #M-MAP-HEADING
```

Figure 3-3 Sample source code for object XX021P01 after Modification.

NEEUEX4: Generate Adapter Name for NJX Conversion

A user exit has been provided, NEEUEX4, which will allow the user to generate an Adapter name for Natural for Ajax conversion depending on the Natural Map Name and allow for the generation of names of status_prop fields based on control variables and fields used in REINPUT MARK. It is available on PC, mainframe and unix platforms.

Typically the user exit would contain similar rules to that used to generate the Adapter name in the Map convertor function in Application Designer.

.

NEEUEX5: Generate Application Names

A user exit has been provided, NEEUEX5, which will allow the user to generate names for Natural Engineer Applications within the repository. It is available on PC, Mainframe and Unix and has to be available to create new Application names.

NEEUEX6: Bespoke Security Interface

A user exit has been provided, NEEUEX6, which will control access to applications when Natural Security does not exist.

The Application name will be passed to the user exit. User will then verify it using whatever method they want.

NB By default if access is restricted then the application name will be marked with a red cross on the treeview main screen. Applications restricted by security may be made invisible to the user by changing the following setting in the initialization file

```
[ENVIRONMENT]
NEETV-SUPPRESS=Y
```

Y = Suppress applications
N = Mark with red cross(Default)

It is available on PC and on Mainframe and Unix when running in a Spod environment. It has to be available.

Application Programming Interfaces

Natural Engineer provides various Application Programming Interfaces (API's) to allow the user to interact their own environments with their Natural Engineer installation.

The API's available are:

1. NEEAPI1N: JCL Flow
2. NEEAPI2N: Extract and Load Services Data
3. NEEAPI3N: Add Links to Services

NEEAPI1N: JCL Flow

NEEAPI1N is an API that allows the user to define JCL Flow scenarios such as those defined within scheduling tools e.g., CA-7 and OPC. The metadata is used within the JCL Flow Chain diagrams.

Prior to adding the metadata into the repository the JCL trigger types should be defined to Natural Engineer. For details on the whole JCL Flow process see the JCL Flow process section in this manual.

For more information on defining JCL Trigger Types please see the Global Properties section of the Natural Engineer Administration manual.

If JCL Flow metadata needs to be deleted from the Natural Engineer repository or overwritten then stand-alone program JCFLDL-P should be executed. This accepts one parameter which is the name of the application.

The API is called with the following parameters, which are detailed in the supplied data area, NEEAPI1A:

```
1 #NEETRG-DATA
 2 #NEETRG-APPLICATION (A8)
 2 #NEETRG-JCL-MEMBER (A32)
 2 #NEETRG-FUNCTION (A15)
 2 #NEETRG-DETAIL (A150)
 2 REDEFINE #NEETRG-DETAIL
  3 #NEETRG-TRG-TYPE (A2)
  3 #NEETRG-TRG-NAME (A44)
  3 #NEETRG-TRG-ID (A10)
  3 #NEETRG-TRG-SETTING (A10)
  3 #NEETRG-TRG-FREQUENCY (A8)
  3 #NEETRG-TRG-FREQUENCY-OPTS (A50)
 2 REDEFINE #NEETRG-DETAIL
  3 #NEETRG-HDR-DATE (A8)
  3 #NEETRG-HDR-TIME (A4)
  3 #NEETRG-HDR-SYSTEM (A8)
  3 #NEETRG-HDR-CPUTM (A5)
  3 #NEETRG-HDR-ELAPTM (A5)
  3 #NEETRG-HDR-ACTIVE (A1)
 2 #NEETRG-ET-REQ (A1)
 2 #NEETRG-STORE-OPPOSITE (A1)
 2 #NEETRG-CREATE-LOG (A1)
 2 #NEETRG-REPLY-DETAIL (A80)
 2 #NEETRG-REPLY (N2.0)
```

PARAMETER	DESCRIPTION
#NEETRG-APPLICATION *	Name of the NEE application that the JCL applies to
#NEETRG-JCL-MEMBER *	Job name initiating the trigger function
#NEETRG-FUNCTION *	HEADER – to store information about the JCL member. TRIGGERED – to store a job triggered by this JCL member. TRIGGEREDBY – to store objects that trigger this JCL member
#NEETRG-DETAIL	Data specific to #NEETRG-FUNCTION (see below)
#NEETRG-ET-REQ	If set to ‘Y’ NEETRG-N will END TRANSACTION every time called. Leave blank to control the timing of END TRANSACTIONs.
#NEETRG-STORE-OPPOSITE	Set to ‘Y’ to create an opposing record to allow both Forward and Backward tracking. Opposite records are unlikely to be necessary when parsing a scheduler output as it is likely to contain both TRIGGERED and TRIGGERBY info.
#NEETRG-CREATE-LOG	If set to ‘Y’ a line is written to PRINT (0) for each record stored. Useful when testing.
#NEETRG-REPLY-DETAIL (output only)	Populated with an error message whenever #NEETRG-REPLY is not 0.

PARAMETER	DESCRIPTION

#NEETRG-REPLY (output only)	0 = Record stored successfully
	1 = Unknown value of #NEETRG-FUNCTION
	2 = Unknown Trigger Type (see below)
	3 = Duplicate Header details found
	4 = Duplicate Trigger details found
	88 = Mandatory data missing
	99 = Unexpected program error from NEEAPI1N
	#NEETRG-REPLY of 0, 1, 2, 88 & 99 should cause termination of calling program.

WHEN #NEETRG-FUNCTION = HEADER	
#NEETRG-HDR-SYSTEM	Name of the system
#NEETRG-HDR-DATE	Last run date. Format should be YYYYMMDD
#NEETRG-HDR-TIME	Time of last run. Format should be HHMM
#NEETRG-HDR-CPUTM	CPU usage. Format of MMMSS
#NEETRG-HDR-ELAPTM	Elapsed time. Format of HHMM
#NEETRG-ACTIVE	Set to ‘Y’ for active jobs
WHEN #NEETRG-FUNCTION = TRIGGERED	
#NEETRG-TRG-TYPE *	This should correspond to a Trigger Type that has been defined in NEE. Trigger Types are added in the Global Properties dialog from the Options menu option.
#NEETRG-TRG-NAME *	Name of the object being triggered by #NEETRG-JCL-MEMBER
#NEETRG-TRG-ID	An ID for this trigger
#NEETRG-TRG-SETTING	A setting for this trigger
#NEETRG-TRG-FREQUENCY	Frequency of the Trigger. Typically when invoked by a calendar within a scheduling tool. e.g. MONTHLY, DAILY. *NB: Usually apply to TRIGGEREDBY recs.*
#NEETRG-TRG-FREQUENCY-OPTS	More frequency info. e.g. MON, WED, FRI. *NB: Usually apply to TRIGGEREDBY recs.*
WHEN #NEETRG-FUNCTION = TRIGGEREDBY	
#NEETRG-TRG-TYPE *	This should correspond to a Trigger Type that has been defined in NEE. Trigger Types are added in the Global Properties dialog from the Options menu option.

PARAMETER	DESCRIPTION
#NEETRG-TRG-NAME *	Name of the object triggering #NEETRG-JCL-MEMBER. This could be another job, a calendar or existence of a particular dataset.
#NEETRG-TRG-ID	An ID for this trigger
#NEETRG-TRG-SETTING	A setting for this trigger
#NEETRG-TRG-FREQUENCY	Frequency of the Trigger. Typically when invoked by a calendar within a scheduling tool. e.g. MONTHLY, DAILY.
#NEETRG-TRG-FREQUENCY-OPTS	More frequency info. e.g., MON, WED, FRI.

* = Mandatory Fields

An example program, NEEAPI1P, is provided that shows sample code to store the relevant JCL Flow data from output for CA-7 scheduler.

NEEAPI1P: Example Program for CA-7

This takes an input file that has been generated by CA-7, interrogates it and then creates records for loading into Natural Engineer via the NEEAPI1N API. The input file would typically be created by the following command in CA-7:

LJOB,LIST=NODD,JOB=*xx**

Where ‘*xx’* is the prefix of the Job Names as specified within CA-7 that you wish to load into Natural Engineer.

The sample program then parses this input file and loads the metadata into Natural Engineer. Information that you may want loaded into Natural Engineer include:

- Last Run Date/Time
- CPUTM – Average CPU time
- ELAPTM – Average Elapsed time
- Scheduled ID

- System Name (NDM,SECS,INFOPAC, etc)
- Active/Inactive

This is purely a sample program for CA-7 and should be tailored to your own environment.

NEEAPI2N: Extract and Load Services Data

NEEAPI2N is an API that allows the user to Extract data from a list of Services & add Services to the NEE Repository.

An example program, NEEAPI2P, is provided that shows how to use it.

NEEAPI2P: Example Program to Extract and Load Services Data

The API is called with the following parameters;

```
01 #NEEAPI2N-PARMS
   02 #NEEAPI2N-SERVICE-NAME       (A180)
   02 #NEEAPI2N-ALT-SERVICE        (A180)
   02 #NEEAPI2N-APPLICATION        (A08)
   02 #NEEAPI2N-OBJECT-NAME        (A32)
   02 #NEEAPI2N-LANGUAGE           (A01)
   02 #NEEAPI2N-OBJECT-TYPE        (A01)
   02 #NEEAPI2N-CALLED-SERVICES     (A) DYNAMIC
   02 #NEEAPI2N-ACTIVE              (A01)
   02 #NEEAPI2N-SERVICE-ID          (A08)
   02 #NEEAPI2N-RESPONSE            (N04)
```

PARAMETER	DESCRIPTION
#NEEAPI2N-SERVICE-NAME	Name of the Service
#NEEAPI2N-ALT-SERVICE	Name of the Alternate Key for the Service
#NEEAPI2N-APPLICATION	If there is an Entry Point for the Service then this is the name of the Application where the Entry Point resides.
#NEEAPI2N-OBJECT-NAME	If there is an Entry Point for the Service then this is the name of the Entry Point object.
#NEEAPI2N-LANGUAGE	If there is an Entry Point for the Service then this is the language of the Entry Point object e.g., N for Natural or C for COBOL.
#NEEAPI2N-OBJECT-TYPE	If there is an Entry Point for the Service then this is the object type of the Entry Point object e.g., N for subprogram or P for Program.
#NEEAPI2N-CALLED-SERVICES	If there are called services then this contains the names of the called services in 180 byte blocks.
#NEEAPI2N-ACTIVE	If the service is active or not. Set to Y or N accordingly.
#NEEAPI2N-SERVICE-ID	This is returned from the API and contains the internal name for the service.
#NEEAPI2N-RESPONSE	This is returned from the API. 0 indicates a successful call to the API.

Note: A service may either have an entry point or called services.

This is purely a sample program and should be tailored to your own environment. Further details are contained with the sample program source provided on the SYSNEE library.

NEEAPI3N: Add Links to Services

NEEAPI3N is an API that allows the user to add links between objects and the services that they call.

An example program, NEEAPI3P, is provided that shows how to use it.

NEEAPI3P: Example Program to Add Links to Services

The sample program needs to identify parameters passed to a trigger object (which may be a stub module for Integration Server Services or a user written object which will call a Service). It does this by interrogating the Natural Engineer repository.

These parameters will comprise the Alternate Key for the Service. The API will then be invoked to attempt to see if the Alternate Key exists, if found a link will be created to the service for the object that called the trigger object. If the Alternate Key is not found then the API will return an invalid response.

Typically the API will be called firstly with an option of DE to delete all existing links followed by option AD to add new ones.

The API is called with the following parameters;

```
01 #NEEAPI3N-STRUCT
  02 #NEEAPI3N-OPTION                  (A02)
  02 #NEEAPI3N-SERVICE-ALT-KEY         (A180)
  02 #NEEAPI3N-TRIGGER-NAME            (A08)
  02 #NEEAPI3N-CALL-LIBRARY            (A08)
  02 #NEEAPI3N-CALL-OBJECT             (A08)
  02 #NEEAPI3N-RESPONSE                (N04)
```

PARAMETER	DESCRIPTION
#NEEAPI3N-OPTION	DE – Delete existing links AD - Add new links
#NEEAPI3N-SERVICE-ALT-KEY	Name of the Alternate Key for the Service. Required for option AD.
#NEEAPI3N-TRIGGER-NAME	Name of the trigger object. Required for option AD.
#NEEAPI3N-CALL-LIBRARY	Set to the library name. Required for option DE and AD.
#NEEAPI3N-CALL-OBJECT	Set to the object name that calls the trigger. Required for option DE and AD.
#NEEAPI3N-RESPONSE	This is returned from the API. 0 indicates a successful call to the API.

This is purely a sample program and should be tailored to your own environment. Further details are contained with the sample program source provided on the SYSNEE library.

3

INDEX