National PDES Testbed
Report Series

The NIST SQL Database Loader: STEP Working Form to SQL

NATIONAL

PD ES

TESTBED

NIST

National PDES Testbed

The NIST SQL Database Loader: STEP Working Form to SQL

Deborah A. Nickerson

NATIONAL PDES TESTBED

NIST

The NIST SQL Database Loader: STEP Working Form to SQL

Deborah A. Nickerson

1 Introduction

This paper discusses the NIST SQL Database Loader; this utility takes a STEP file and populates SQL tables with instances of STEP entities. This document is intended as a programmer and user's guide. After reading Translating Express to SQL: a User's Guide, users may skip to the User's Manual section (section 5.0). Programmers should read the following prerequisite reading materials before this document: The NIST Working Form for STEP, The NIST STEP Working Form Programmer's Reference, Translation of Express Schema into SQL, and Translating Express to SQL: a User's Guide. In addition, objects in the STEP Working Form are tightly coupled with the Express Working Form; some familiarity with the latter will be assumed (see Fed-X: The NIST Express Translator and NIST Express Working Form Programmer's Reference).

The NIST SQL Database Loader is implemented in ANSI Standard C [ANSI89b]. The Oracle Pro*C precompiler is used to embed SQL standard queries in accordance with the ANSI standard for Embedded SQL [ANSI89c].

1.1 Context

The Standard for the Exchange of Product Model Data (STEP) is an emerging standard for the interchange of product data between various vendors' CAD/CAM systems and other manufacturing-related software. A National PDES Testbed has been established at the National Institute of Standards and Technology to provide testing and validation facilities for the emerging standard. The Testbed is funded by the CALS (Computer-aided Acquisition and Logistic Support) program of the Office of the Secretary of Defense. As part of the testing effort, NIST is charged with providing a software toolkit for manipulating STEP data. This NIST PDES Toolkit is an evolving, research-oriented set of software tools. This document is one of a set of reports which describe various aspects of the Toolkit. An overview of the Toolkit is provided in An Introduction to The NIST PDES Toolkit, along with references to the other documents in the set.

For further information on the NIST SQL Database Loader or other components of the Toolkit, or to obtain a copy of the software, use the attached order form.

2 Architecture

As discussed in The NIST Working Form for STEP, STEPparse consists of two separate passes: parsing and output generation. During the first pass, the Express file is

parsed, and the Express Working Form is created. Using the Express Working Form, the STEP file is parsed, and an instance of a `Product` Data Definition is built in the STEP Working Form. The second pass is the linking of the output module, in this case, the NIST SQL Database Loader. The NIST SQL Database Loader can be dynamically or statically linked. Once linked, the NIST SQL Database Loader, following through the `Product` Data instance in the STEP Working Form, loads the `Product` information into the appropriate database tables.

2.1 Makefile

The makefile used by the NIST SQL Database Loader is the generic makefile used by all the modules that form the Validation Testing System Toolkit. Hooks for the Oracle Pro*C precompiler were added.

2.2 Database Schema

As discussed in <u>Translating Express to SQL: a User's Guide</u>, an Express schema is mapped into a relational database schema. This schema creates the necessary tables needed by each entity in the Express schema and dictionary tables to store Express semantic construct information. In addition to the above dictionary tables, the NIST SQL Database Loader uses a cross referencing table, `sys$entityid_tableid`. This table consists of three columns: `entityid`, `tableid`, and `fileid`, representing the STEP file entity ID of an `object`, the table ID of an `object`, and the file number the entity instance is in. Figure 1 shows an example STEP file `objects` and their corresponding entries in the `sys$entityid_tableid` table. The `sys$entityid_tableid` table is created at the same time the other data dictionary tables are created, but is only populated by the NIST SQL Database Loader.

```
STEP file objects.

    @1=DIRECTION(,,0.,0.,1.0000000308274466);
    @2=DIRECTION(,,1.0000000308274466,0.,0.);
    @3=DIRECTION(,,0.,1.0000000308274466,0.);
    @4=CARTESIAN_POINT(,,0.20.428009033203125,11.22300052
    64282227);
```

Representation of table SYS$ENTITYID_TABLEID in database.

SYS$ENTITYID_TABLEID

ENTITYID	TABLEID	FILEID
1	DIRECTION!00000000	NULL
2	DIRECTION!00000001	NULL
3	DIRECTION!00000002	NULL
4	CARTESIAN_POINT!00000003	NULL

Figure 1: STEP File objects & SYS$ENTITYID_TABLEID Table

3   `stepwf_sql` Internals

The NIST SQL Database Loader's program name is `stepwf_sql`. As with other STEP Working Form output modules, `stepwf_sql` is linked to STEPparse via an `entry_point` function with the following parameters: `product` and `file`, corresponding to an instance of a Product Data Definition in the STEP Working Form and the output file respectively. Unlike other STEP Working Form output modules, the output file is not used as input data, rather, it is used for debugging and informational purposes only.

3.1   `stepwf_sql` Flow Control

Moving through the `product` structure, `stepwf_sql` extracts one by one the `objects` contained within the `product` structure. Each new `object` signals the start of a new SQL insert statement. If the `object` is sharable by other `objects`, an entry is made into the `sys$entityid_tableid` table. The attributes of the `object` are extracted, and each attribute's type is checked.

For *simple (Express base)* type attributes, such as integers, the attribute's value is appended to the current `object`'s dynamic SQL insert statement.

For *entity* type attributes, the attribute is determined to be either embedded or not embedded. For non-embedded entity attributes, a query is made to the `sys$entity-`

`id_tableid` table for the table ID of the referenced entity ID; the table ID is appended to the current `object`'s dynamic SQL insert statement. For embedded entity attributes, a recursive process is started where the entity attribute is treated as a new `object`.

For *select* type attributes, the value is retrieved, then its type is tested. If the value's type is that of an entity, a recursive process is started treating the value of the select as a new `object`. Currently, the *Express* to *SQL* mapping, described in <u>Translating Express to SQL: a User's Guide</u>, states that if the type of the value of the select is not entity, then, the value is represented as a string and appended to the current `object`'s dynamic SQL insert statement.

For *aggregate* type attributes, a new dynamic SQL insert statement is assembled for each element of the aggregate. If the element of the aggregate is of type aggregate, a recursive approach is taken, adding on subscripts to the newly created dynamic SQL insert statement until a simple element is found. Figure 2 shows an example of the B_SPLINE_CURVE aggregate attribute KNOTS.

STEP file `object`: B_SPLINE_CURVE. The KNOTS attribute is highlighted.

@370=B_SPLINE_CURVE(,,4,8,
(#310,#320,#325,#330,#340,#350,#360,#365,#368),
.UNIFORM_KNOTS.,3,(5,4,5),*(0.0,1.0,2.0)*,
(1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0),,,.F.);

B_SPLINE_CURVE aggregate attribute KNOTS table.

B_SPLINE_CURVE#KNOTS

ID	SUBSCRIPT_1	VALUE
B_SPLINE_CURVE!00000381	0	0.0
B_SPLINE_CURVE!00000381	1	1.0
B_SPLINE_CURVE!00000381	2	2.0
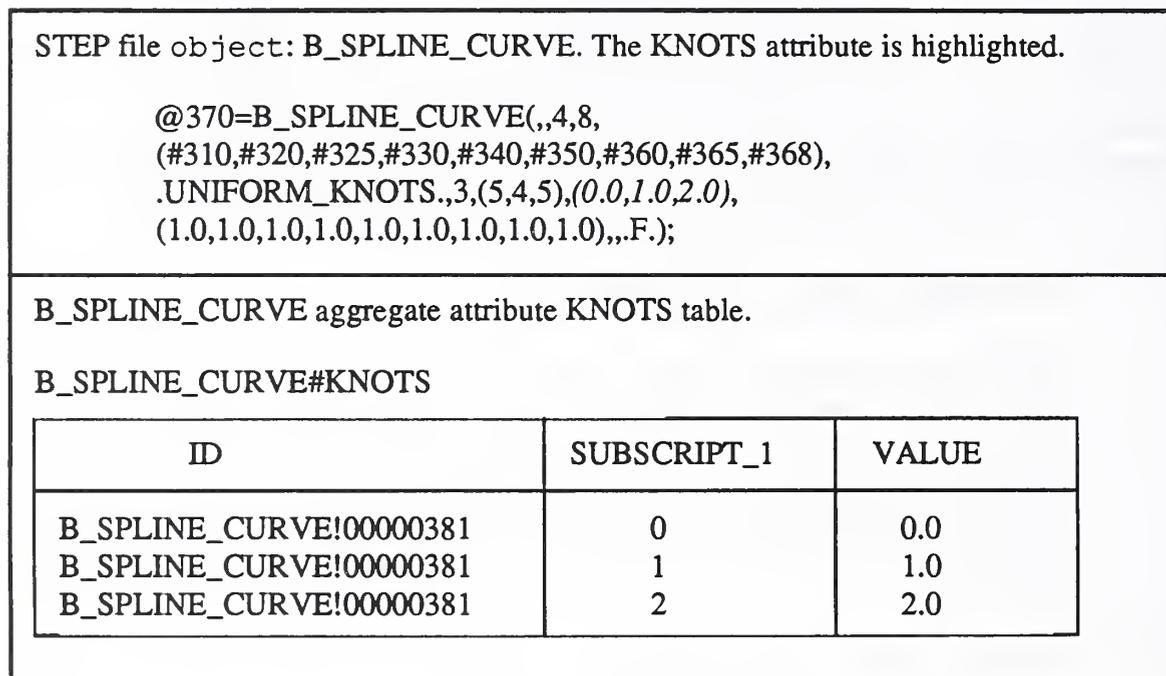
Figure 2: Aggregates

Once a dynamic SQL statement is completely assembled, it is inserted into the database; this is done until the `product` is completely loaded.

3.2   Working Form Routines

`stepwf_sql` extracts the product information in the STEP Working Form through a series of STEP Working Form access functions; these are listed in <u>The NIST STEP Working Form Programmer's Reference</u>. In addition, access functions to the Express

Working Form are also used; these can be found in <u>The NIST Express Working Form Programmer's Reference</u>.

3.3 Embedded Pro*C

The NIST SQL Database Loader runs interactively with the Oracle database, using embedded Pro*C code. Pro*C code is used throughout `stepwf_sql` to connect to the user's database account, extract dictionary information, and of course, insert the part data. Dynamic SQL is used to insert the different STEP file entities of the part into their respective tables.

3.4 Table ID Assembly

To insure that each table ID is distinct, the following method for creating table IDs was devised. Given an `object`'s Express entity name, the `EXPRESSYS$NAMES` table is queried for the Oracle entity name. An exclamation point (!) is appended to the Oracle entity name, and finally a number is appended. This number is generated by an Oracle sequence, `TABLE_SEQ`; this sequence is created upon database schema creation.

3.5 Error Checking

Extensive error checking is included in `stepwf_sql`. Since the Express and STEP parsers are executed before the NIST SQL Database Loader, any errors and/or warnings in the Express and STEP files should be flagged before reaching this software. On the other hand, there may be errors in the database. For this reason, `stepwf_sql` will report errors it finds in a descriptive manner; for example, if there is no table in the database that matches the `object` to be inserted, a *rollback* of the database is executed, and a descriptive error message will be displayed including the entity ID of the `object` where execution failed. In addition, the output file generated lists all the error free inserts made into the database before *rollback*.

4 Issues of Concern

Currently, `stepwf_sql` has the ability to load in a partial STEP file, then at a later point in time, load the subsequent data. It does not have the ability to load more than one part in the database. In order to load multiple parts into the same database, the issue of how to differentiate parts needs to be resolved. At the present time, the PSCM structures would have to be populated to differentiate parts; there is nothing explicit in the STEP file that differentiates parts.

5 User's Manual

This section deals with the "how to" of database loading. It assumes that the user has already created a database from the Express schema; look at the <u>Translating Express to SQL: User's Guide</u> for instructions.

5.1   Command Line

The command line for stepwf_sql is as follows:

```
% stepwf_sql -e express_schema.file -s step.file
```

The Express schema file used should be the same as the one used to build the database schema, and the STEP file to be loaded should match the schema exactly, i.e. all attributes and references should follow the structure of the schema.

First, the Express file is parsed, building the Express Working Form. No errors of detrimental importance to the database schema should be reported at this time; these should have been dealt with at schema creation.

Second, the STEP file is parsed, building the STEP Working Form. Any errors should be noted with great care. The error messages from the STEP parser are self-explanatory. If error messages are present and the program has not stopped processing, abort the process, and fix the errors. stepwf_sql will have to be restarted.

If no error messages are reported, the user will be asked for their account user name and password. They will also be asked to fill in an output file name; this output file is for informational and debugging purposes only. Error messages after this point indicate a problem with the user's database. These messages are also straightforward, for ease in debugging. Once the errors are corrected, stepwf_sql will need to be restarted; stepwf_sql *rollbacks* the database to the state prior to execution.

5.2   Useful Database Utilities

There are a number of useful database manipulation utilities available; descriptions of these can be found in The PDES Testbed User's Guide. Of these utilities, the utility userdropdata drops all the data contained in a user's database. As mentioned in the above "Issues of Concern" section, the database can only hold one part at a time. Each time the user wants to load a new STEP file, the user must clear the database using the userdropdata utility.

6   Conclusion

The NIST SQL Database Loader takes a STEP file and Express schema as input. Express schema information and STEP file data are loaded into memory working forms. Data is retrieved from the memory resident working forms and used to populate SQL tables with instances of STEP entities.

For further information on the STEP Working Form, the database schema, or other components of the Toolkit, or to obtain a copy of the software, use the attached order form.

A   References

[Clark90a]   Clark, S. N., <u>An Introduction to The NIST PDES Toolkit</u>, NISTIR 4336, National Institute of Standards and Technology, Gaithersburg, MD, May 1990

[Clark90b]   Clark, S.N., <u>Fed-X: The NIST Express Translator</u>, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD, forthcoming

[Clark90c]   Clark, S.N., <u>The NIST Working Form for STEP</u>, NISTIR 4351, National Institute of Standards and Technology, Gaithersburg, MD, June 1990

[Clark90e]   Clark, S.N., <u>NIST Express Working Form Programmer's Reference</u>, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD, forthcoming

[Clark90f]   Clark, S.N., <u>NIST STEP Working Form Programmer's Reference</u>, NISTIR 4353, National Institute of Standards and Technology, Gaithersburg, MD, June 1990

[Metz89]   Metz, W.P., and K.C. Morris, <u>Translation of an Express Schema into SQL</u>, PDES Inc. internal document, November 1989

[Morris90]   Morris, K.C., <u>Translating Express to SQL: a User's Guide</u>, NISTIR 4341, National Institute of Standards and Technology, Gaithersburg, MD, May 1990

[Strouse90]   Strouse, Kathleen, and Michael McLay, <u>PDES Testbed User's Guide</u>, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD, forthcoming

[ANSI89a]   American National Standards Institute, <u>Database Language SQL with Integrity Enhancement</u>, Document ANSI X3.135-1989

[ANSI89b]   American National Standards Institute, <u>Programming Language C</u>, Document ANSI X3.159-1989

[ANSI89c]   American National Standards Institute, <u>Database Language - Embedded SQL</u>, Document ANSI X3.168-1989

MAIL TO:
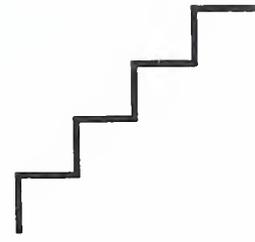
NATIONAL

PDES
ES

TESTBED

National Institute of Standards and Technology

Gaithersburg MD., 20899

Metrology Building, Rm-A127

Attn: Secretary National PDES Testbed

(301) 975-3508

Please send the following documents and/or software:

☐ Clark, S.N., An Introduction to The NIST PDES Toolkit

☐ Clark, S.N., The NIST PDES Toolkit: Technical Fundamentals

☐ Clark, S.N., Fed-X: The NIST Express Translator

☐ Clark, S.N., The NIST Working Form for STEP

☐ Clark, S.N., NIST Express Working Form Programmer's Reference

☐ Clark, S.N., NIST STEP Working Form Programmer's Reference,

☐ Clark, S.N., QDES User's Guide

☐ Clark, S.N., QDES Administrative Guide

☐ Morris, K.C., Translating Express to SQL: A User's Guide

☐ Nickerson, D., The NIST SQL Database Loader: STEP Working Form to SQL

☐ Strouse, K., McLay, M., The PDES Testbed User Guide

OTHER (PLEASE SPECIFY)

_____

_____

_____

_____

These documents and corresponding software will be available from NTIS in the future. When available, the NTIS ordering information will be forthcoming.

NIST

4. TITLE AND SUBTITLE

The NIST SQL Database Loader: STEP Working Form to SQL

5. AUTHOR(S)

Deborah A. Nickerson

11. ABSTRACT (A 200-WORD OR LESS FACTUAL SUMMARY OF MOST SIGNIFICANT INFORMATION. IF DOCUMENT INCLUDES A SIGNIFICANT BIBLIOGRAPHY OR LITERATURE SURVEY, MENTION IT HERE.)

The Standard for the Exchange of Product Model Data (STEP) is an emerging standard for the exchange of product information among various manufacturing applications. A SQL database containing product data definitions has been provided as a facility for testing STEP. The National PDES Testbed at NIST has developed software to load a SQL database with instances of a product data definition.

12. KEY WORDS (6 TO 12 ENTRIES; ALPHABETICAL ORDER; CAPITALIZE ONLY PROPER NAMES; AND SEPARATE KEY WORDS BY SEMICOLONS)

EXPRESS, National PDES Testbed, PDES, Product Data Exchange, SQL, STEP