The NIST EXPRESS Toolkit
–
Introduction and Overview

Don Libes

Abstract

The NIST EXPRESS Toolkit is a software library for building EXPRESS-related tools. This paper gives an introduction, overview, and history of the toolkit. This paper also describes how to get more information on the toolkit. No knowledge of EXPRESS or the EXPRESS Toolkit is presumed other than a rudimentary grasp of basic computer science.

Keywords: compiler, EXPRESS; history; implementation; National PDES Testbed; PDES; STEP

Context

The PDES (Product Data Exchange using STEP) activity is the United States' effort in support of the Standard for the Exchange of Product Model Data (STEP), an emerging international standard for the interchange of product data between various vendors' CAD/CAM systems and other manufacturing-related software [10]. 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 Computer-aided Acquisition and Logistic Support (CALS) program of the Office of the Secretary of Defense.

As part of the testing effort, NIST is charged with providing software for manipulating STEP data. The NIST EXPRESS Toolkit is a part of this. The toolkit is an evolving, research-oriented set of software tools. This document is one of a set of reports ([1] – [9]) which describe various aspects of the Toolkit.

Introduction

The NIST EXPRESS Toolkit is a software library for building software tools for manipulating information models[1] written in the EXPRESS language [11]. An example application ("fedex") is included which reports syntactic and semantic errors in EXPRESS schemas.

Figure 1 shows the toolkit in context. The toolkit acts as a database for schema information stored in the file system. Using the toolkit, an application can query for information about the schema such as "What entities are defined?" (Listing 1) or "What are the attributes in entity CURVE?"(Listing 2). The application can also manipulate or augment information in the toolkit.

1. The terms *information model* and *conceptual schema* are used interchangeably throughout this document.

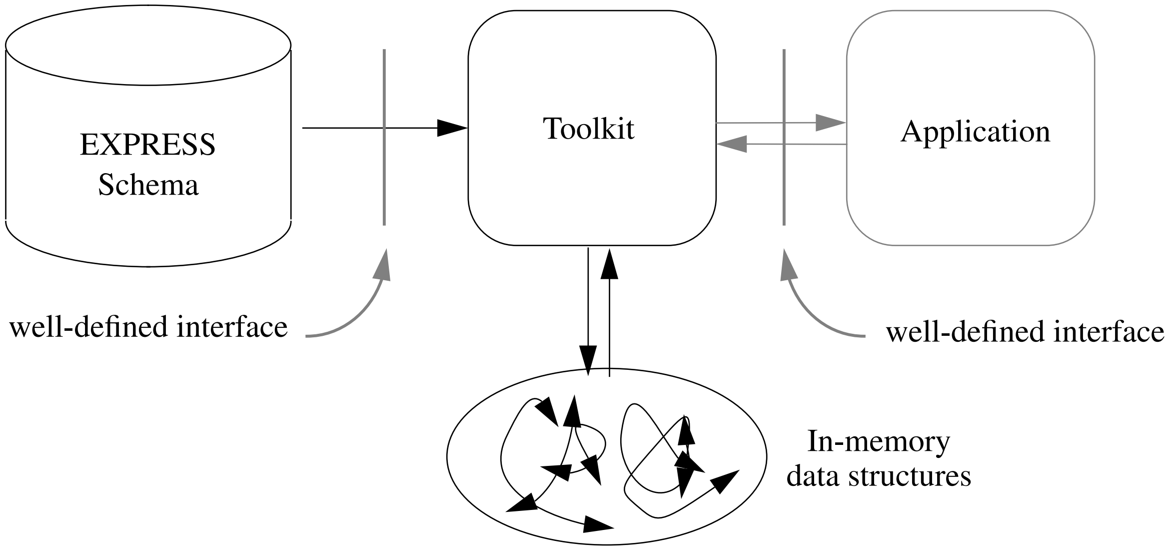

Figure 1: Data flow in the toolkit

```
LISTdo(SCOPEget_entities(schema),e,Entity)
      printf(ENTITYget_name(e));
LISTod
```

Listing 1: What entities are defined in a schema?

```
LISTdo(ENTITYget_attributes(entity),v,Variable)
      printf(VARget_name(v));
LISTod
```

Listing 2: What attributes are defined in an entity?

The actual in-memory data structures used by the toolkit are irrelevant to the application since the toolkit provides a well-defined interface for access to all information. This well-defined interface is a set of function calls encapsulating the information contained in the schema.

The toolkit allows tools to be schema-independent. Different schemas can be read at run-time, allowing applications to be flexible in the data that they manipulate. For example, we have built a Part 21 exchange file parser [12] that works with any EXPRESS schema. Another class of such applications is translators. We have built translators to convert from EXPRESS to C++, SmallTalk, and SQL.

The translators, in turn, may be used to build other applications which are schema-dependent. We have built schema-dependent tools such as the Data Probe, prototype EXPRESS and SQL schema browsers, and data editors [13][14]. It is important to recognize that similar applications while schema-dependent, did not actually require the developer to write any schema-dependent code. Such code was produced by the translators, and can be reproduced for other schemas since the translators themselves are schema-independent.

The choice between schema-dependent and schema-independent depends on several factors. Schema-independent applications usually are physically smaller since they do not have entire information models embedded within them. These applications are also insulated against changes in the conceptual schema and, to a certain extent, in EXPRESS itself.

On the other hand, schema-dependent applications invariably have less overhead during run-time devoted to initialization of the in-memory representation of the information model, and can often have reduced time for data access as well.

We have also constructed other toolkits that work with the EXPRESS Toolkit. For example, we have built Exppp, a toolkit for pretty-printing (i.e., formatting) EXPRESS [15]. This exists as a separate toolkit since it represents just one style of formatting and there could conceivably be many others. The Exppp Toolkit along with the EXPRESS Toolkit has been used to build several more tools such a program to convert STEP Short Listings to Annotated Listings [16], and a program to manipulate EXPRESS within a Tcl/Tk environment [17].

Like tools, toolkits can similarly be either schema-dependent or schema-independent. The Exppp toolkit is schema-independent. Another schema-independent toolkit is the STEP Class Library (SCL) Toolkit [18] which provides support for manipulating EXPRESS inside a C++ environment. In contrast, the Part 21 Exchange File Toolkit [12] is schema-independent.

Environment

The EXPRESS Toolkit was developed on Sun Microsystems SPARCstation workstations running SunOS, an operating system derived from BSD UNIX.[2] Occasionally, we or others have ported the software to other platforms such as Digital's DECstation and Hewlet-Packard's HP700 and 800-series workstations. While some small non-portabilities invariably creep in, the system is highly portable within a UNIX environment. While we have not tried doing so, we believe the software will port easily to any pure-POSIX [19] environment. With minor limitations due to file system deficiencies, the software should be portable to a PC-based platform.[3]

The Toolkit is written in ANSI Standard C [20]. The use of prototypes prevents its use in pure K&R [21] environments although this could be remedied by code rewriting tools. While any ANSI C compiler can be used to compile the toolkit, we use GCC, available from the Free Software Foundation (FSF) [22]. FSF tools are free but with certain distribution restrictions. If GCC is used to compile the toolkit, certain optimizations are enabled which can increase its performance.

The toolkit's scanner is written in Lex [23], a scanner generator commonly provided in UNIX environments. However, it can also be processed by Flex, another scanner generator that is in the public-domain and quite popular. The toolkit's parser is written in Yacc [24] (a parser generator commonly provided in UNIX environments) with special modifications [25] to enhance error reporting. However, it can also be processed by Bison, another parser generator that is available from FSF. While not built in to UNIX systems, Flex and Bison are more flexible than Lex and

2. Trade names and company products are mentioned in the text in order to adequately specify experimental procedures and equipment used. In no case does such identification imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the products are necessarily the best available for the purpose.

3. Previous releases were dependent upon a POSIX.2 environment. We have removed these dependencies.

Yacc, and they have a reputation of being faster as well. Although we have not benchmarked them, we use Flex and Bison in our own development workbench and encourage their use in the toolkit.

Conformance

On September 8, 1993, the CAD/CAM Data Exchange Technical Centre (CADDETC), a British Standards Institution (BSI) accredited testing body of the United Kingdom, certified that the NIST EXPRESS Toolkit had passed CADDETC's Level 1 (reference and syntax checking) conformance testing. As of this writing, the toolkit is the only software to have achieved this.

Performance

Performance has not been objectively studied, however we can say something about it nonetheless. The performance of the toolkit has been significantly improved from earlier releases. The current release runs in less than 1% of the time than the previous release [26] and uses 60% less space, while at the same time semantically analyzing more of the information in the EXPRESS specification than before. The area of performance is further described in [3].

While there is no standard schema or platform with which we can characterize performance, some simple statistics are possible. On a Sun SPARCstation 2, a 100Kb schema takes on the order of 1 to 2 seconds to process including reporting any syntactic or semantic errors. The toolkit is approximately 14000 lines of C (including comments) which compiles in 2 minutes using GCC.

How to Obtain the Toolkit

The toolkit and its documentation may be obtained in a variety of ways. The simplest way is through anonymous ftp via the Internet. In this case, the source is pub/step/npttools/exptk.tar.Z on ftp.cme.nist.gov. Complete documentation on obtaining the toolkit and its documentation is in /pub/step/ntpdocs/exptk-obtaining-installing.ps.Z [5].

Alternately, it is possible to receive the toolkit by email. To do this, send the following mail to ntpserver@cme.nist.gov:

```
send step/npttools/exptk.tar.Z
send step/nptdocs/exptk-obtaining-installing.ps.Z
```

If you do not understand these instructions or for any other reason cannot successfully use ftp or email, contact:

FASD – National PDES Testbed
National Institute of Technology and Standards
Bldg 220, Rm A-127
Gaithersburg, MD 20899

npt-info@cme.nist.gov
1-301-975-3179

Questions, Problems, and Support

The system is distributed in source form and you are encouraged to experiment with the system, especially if you have problems with it. While it is often quicker for you to have us diagnose your problems, it is quicker for us to have you diagnose your own problems. This software is a prototype, intended to spur development of commercial products.

Nonetheless, if you do have questions and/or problems, you may send e-mail to the following addresses. Please include schemas, version numbers, platform descriptions, and any other information that could be relevant.

Annotated Listing Generator	**shtolo@cme.nist.gov**
Data Probe	**dprobe@cme.nist.gov**
EXPRESS Analysis	**exptk@cme.nist.gov**
EXPRESS Server	**express-server-admin@cme.nist.gov**
Part 21 Analysis	**p21tk@cme.nist.gov**
STEP Class Library	**stepcl@cme.nist.gov**

History and Credits

The idea of a schema-independent toolkit was first proposed by Steve Clark (NIST). Clark wrote the initial release of the toolkit. Written in C, it was a non-object-oriented implementation characterized by a single, single-pass-resolution phase. It was based on the "Tokyo" draft of EXPRESS.

After attempting a short-lived version in C++, Clark rewrote it in C but with a hand-built object-oriented engine for N496 [27]. This was publicly released in 1988 and saw wide distribution.

Around the same time, Bruce Thomas (NIST) created a similar toolkit for what was to become the STEP Part 21 Exchange File Format. Several other NIST employees worked on this including Sandy Ressler, Tina Lee, and Cathy Diaz. Clark eventually took over control of this software, integrating it into a framework similar to the EXPRESS toolkit. Using both toolkits, Clark wrote the first application, an EXPRESS to Smalltalk translator [28]. In the following year, numerous applications appeared, including an EXPRESS to SQL translator and an EXPRESS to C++ translator, both written by KC Morris (NIST).

In 1989, Clark began participating in the EXPRESS standards committee, while relinquishing further software development to Don Libes (NIST). Libes worked on speeding up the implementation primarily by reimplementing symbol tables with hash tables instead of linked lists. In September 1990, based on N496, this release was distributed to NIST's PDES, Inc. partners but was not made publicly available.

Libes then enhanced the software so that it supported N14. Dave Briggs (Boeing and PDES, Inc.) contributed the implementation of Use and Reference. This implementation was publicly distributed in November 1991.

Up to this point, this and other software was collectively known as the "NIST PDES toolkit". As different rates of revision in various standards caused pieces of the software to be revised separately, the PDES toolkit was broken up into a number of toolkits, such as the EXPRESS Toolkit.

During 1992, Libes rewrote the EXPRESS Toolkit while ostensibly converting it from N14 to N151 (Draft International Standard). In the interests of efficiency, the object-oriented engine was removed, and the single-pass resolution was converted to multiple passes. This software ran over 100 times faster than the earlier object-oriented releases. In addition, many of the missing features of EXPRESS were finally implemented. This is described further in [3].

The authors thank the numerous testers and application writers who put up with continual "improvements" to the toolkit, and who gave high-quality feedback. Thanks particularly to Jim Fowler, KC Morris, Kent Shepherd, Gerard Silvernale, Tom Kramer, Kent Reed, Gerard Silvernail, Connie Augustin, Newton Breese, Cita Furlani, Lisa Phillips, Dave Sauder, Mary Mitchell, Peter Carr, David Helfrick, Sandy Ressler, Jeane Ford and numerous other people who contributed requirements, suggestions, and bug fixes.

The NIST EXPRESS Toolkit is funded by the Computer-aided Acquisition and Logistic Support (CALS) program of the Office of the Secretary of Defense (see Context on page 1).

Documentation

The following papers describe various aspects of the toolkit. Note that the implementation is subject to change. Because of this, no guarantee is made that the descriptions in this paper are still accurate with regard to the current implementation.

[1] Libes, Don, and Fowler, Jim, "The NIST EXPRESS Toolkit – Requirements", NISTIR 5212, National Institute of Standards and Technology, Gaithersburg, MD, June 9, 1992.

This describes the original requirements that provided the impetus for the design of the current release of the toolkit.

[2] Libes, Don, "The NIST EXPRESS Toolkit – Design and Implementation", *Proceedings of the Seventh Annual ASME Engineering Database Symposium*, San Diego, CA, August 9-11, 1993.

This describes the design and implementation of the toolkit.

[3] Libes, Don, and Clark, Steve, "The NIST EXPRESS Toolkit – Lessons Learned", *Proceedings of the 1992 EXPRESS Users' Group (EUG '92) Conference*, Dallas, Texas, October 17-18, 1992.

This describes implementation experiences while building the toolkit. It describes much of the rational for the architectures used in the various releases, and why they changed.

[4] Libes, Don, "The NIST EXPRESS Toolkit – Using Applications", NISTIR 5206, National Institute of Standards and Technology, Gaithersburg, MD, June 9, 1993.

This describes fedex, the included toolkit application to check EXPRESS syntax and semantic errors. Its options and diagnostics are described. The EXPRESS toolkit mail server is also described.

[5] Libes, Don, "The NIST EXPRESS Toolkit – Obtaining and Installing", NISTIR 5204, National Institute of Standards and Technology, Gaithersburg, MD, June 9, 1993.

This describes how to obtain and install the EXPRESS toolkit.

[6] Libes, Don, "The NIST EXPRESS Toolkit – Programmer's Reference", National Institute of Standards and Technology, Gaithersburg, MD, to appear.

This describes the toolkit's application interface.

[7] Libes, Don, "The NIST EXPRESS Toolkit – Creating Applications", National Institute of Standards and Technology, Gaithersburg, MD, to appear.

This describes how to design and implement toolkit applications. fedex_plus and the Part 21 parser are used as examples.

[8] Libes, Don, "The NIST EXPRESS Toolkit – Updating Existing Applications", NISTIR 5205, National Institute of Standards and Technology, Gaithersburg, MD, June 9, 1993.

This describes how to update applications that used the previous toolkit.

[9] Libes, Don, "The NIST EXPRESS Server – Usage and Implementation", National Institute of Standards and Technology, Gaithersburg, MD, to appear.

This describes a server which runs the applications based on the NIST EXPRESS toolkit. It is possible to use the server via internet mail or telnet/rlogin, thereby avoiding the overhead of installing and maintaining the tools locally.

Other References

[10] Mason, H., ed., "Industrial Automation Systems – Product Data Representation and Exchange – Part 1: Overview and Fundamental Principles", Version 9, ISO TC184/SC4/ WG PMAG Document N50, December 1991.

[11] Spiby, P., ed., "ISO 10303 Industrial Automation Systems – Product Data Representation and Exchange – Part 11: Description Methods: The EXPRESS Language Reference Manual", ISO DIS 10303-11:1992(E), July 15, 1992.

[12] Libes, Don, "The NIST STEP Part 21 Exchange File Toolkit – An Update", NISTIR 5187, National Institute of Standards and Technology, Gaithersburg, MD, September 8, 1993.

[13] Morris, K.C., "Data Probe: A Tool for EXPRESS-based Data", *Proceedings of the Seventh Annual ASME Database Symposium - Engineering Data Management: Key to Success in a Global Market*, American Society of Mechanical Engineers, New York, August 1993.

[14] Morris, K.C., "Translating EXPRESS to SQL: A User's Guide", NISTIR 4341, National Institute of Standards and Technology, Gaithersburg, MD, May 1990.

[15] Libes, Don, "Exppp – An EXPRESS Pretty Printer, to appear.

[16] Libes, Don, "Shtolo – Converting STEP Short Listings to Annotated Listings, to appear.

[17] Ousterhout, John K., *Tcl and the Tk Toolkit*, Addison-Wesley, 0-201-63337-X, February, 1994.

[18] McLay, M.J., Morris, K.C., "The NIST STEP Class Library", *C++ at Work-'90 Conference Proceedings*, reprinted as NISTIR 4411, September 1990.

[19] Federal Information Processing Standard 151, *POSIX: Portable Operating System Interface for Computer Environments*, IEEE 1003.1/Draft 12, September 1988.

[20] American National Standards Institute, *Programming Language C*, Document ANSI X3.159-1989.

[21] Kernighan, Brian, and Ritchie, Dennis, *The C Programming Language*, Prentice Hall, New York, NY, 1978.

[22] Stallman, Richard M., et al, *GNU's Bulletin*, Free Software Foundation, Inc., Cambridge, MA, June 1992.

[23] Lesk, M.E. and Schmidt, E., Lex: A Lexical Analyzer Generator, *UNIX Programmer's Manual*, Seventh Edition, Bell Laboratories, Murray Hill, NJ, 1978.

[24] Johnson, S.C., "Yacc: Yet Another Compiler compiler", *UNIX Programmer's Manual*, Seventh Edition, Bell Laboratories, Murray Hill, NJ, 1978.

[25] Schreiner, Axel T. and Friedman, Jr., H. George, *Introduction to Compiler Construction with UNIX*, New York, NY, Prentice Hall, 1985.

[26] Clark, Steve N., "An Introduction to The NIST PDES Toolkit", NISTIR 4336, National Institute of Standards and Technology, Gaithersburg, MD, May 1990.

[27] Schenck, D., ed., "Exchange of Product Model Data - Part11: The EXPRESS Language", ISO TC184/SC4 Document N496, July 1990.

[28] Clark, Steve N., "QDES User's Guide", NISTIR 4361, National Institute of Standards and Technology, Gaithersburg, MD, June 1990.