DRAFT - Technical Report CAML/CSL
December 6, 1995

Guidelines for the Evaluation of Electronic Data Interchange Products

John J. Garguilo
Paul Markovitz

U. S. Department of Commerce
National Institute of Standards and Technology
Computing and Applied Mathematics Laboratory
Computer Systems Laboratory
Gaithersburg, MD 20899, USA

Contents

List of Figures

List of Tables

1 Introduction

The need to exchange information is critical within the business community. The information may be generic in nature, such as a purchase order or invoice, or specific to an organization, such as a customs declaration. Traditionally, companies have exchanged this information by mailing preprinted business forms. By integrating computers and data communications into the business process, companies can reap the benefits of exchanging information electronically: reduced paperwork, minimized cost, and improved response time. This process, the computer-to-computer exchange of standardized business information, is called Electronic Data Interchange (EDI).

Businesses have three options for implementing an EDI system.

1. Businesses can develop their own EDI software. This option is expensive and time-consuming, and since there is risk involved with developing new software, this option is discouraged except in extreme cases (e.g., using a hardware platform for which no commercial EDI software is available).

2. Businesses can use an EDI service bureau (e.g., a third-party network). With this option, a company sends its business transactions to the service bureau which performs the EDI service at its location. Since the fees for this service are generally high, service bureaus are only recommended as a short-term solution.[1]

3. Businesses can purchase an EDI product. Product purchase is usually the most cost-effective, long-term solution for implementing an EDI system.

As with most software products, EDI products can differ greatly. Some products provide communications functions. Some products include forms generation and data entry capabilities. The host of options potentially present in an EDI product can make purchasing the *right* one a difficult task. By understanding the fundamentals of EDI software, however, one can perform an intelligent evaluation of commercial EDI products.

The primary objective of this document is to assist the reader in determining which EDI product, among many candidate products, best meets the reader's requirements. To achieve this objective, the following chapters are provided. Chapter 2 is an EDI tutorial. Chapter 3 describes functionality potentially present in any EDI product. Chapter 4 discusses issues regarding the performance of EDI products. Chapter 5 addresses issues relating to integrating EDI products into the business process, and chapter 6 concludes the paper.

Three appendices are also provided with this paper. Appendix A overviews the Electronic Commerce Integration Facility (ECIF), an electronic commerce program initiated by the National Institute of Standards and Technology (NIST). Appendix B lists the vendors participating in the ECIF procurement testbed, and appendix C presents the data used by NIST to test the performance of EDI translators.

[1] Not every business will want EDI capabilities on-site. For such a business an EDI service bureau can convert EDI documents to a format acceptable to its business application, or to a FAX if the business does not have computing capability.

2 EDI Tutorial

The functional and performance evaluation chapters of this paper assume the reader has a basic understanding of EDI. So as not to limit the audience of this paper to those familiar with EDI, this tutorial is provided. The tutorial introduces many EDI terms and concepts, serving as a learning utility for the novice EDI user, and as a comprehensive review for those more experienced with EDI.

2.1 The Problem Addressed by EDI

Companies have used paper as the traditional medium for conducting business. Company records are filed on paper, and paper forms are mailed between companies to exchange information. The advent of the business computer has enabled companies to process data electronically; however, the exchange of this data between companies still relies heavily on the postal system. Oftentimes, a company will enter data into a business application, print a form containing the data, and mail this form to a trading partner. The trading partner, after receiving the form, re-keys the data into another business application. Inherent in this process are poor response times - use of the postal system can add days to the exchange process, excessive paperwork for both companies involved in the exchange, and the potential for errors as information is transcribed.

2.2 History of EDI

Although the business computer enabled companies to store and process data electronically, companies needed an expedient method to communicate the data. This method was realized by the widespread use of computer telecommunications. Using telecommunications, companies could transmit data electronically over telephone lines, and have the data input directly into a trading partner's business application. These electronic interchanges improved response time, reduced paperwork, and eliminated the potential for transcription errors. Computer telecommunications, however, only solved part of the problem.

Early electronic interchanges were based on proprietary formats agreed between two trading partners. Due to differing document formats, it was difficult for a company to exchange data electronically with many trading partners. What was needed was a *standard* format for the data being exchanged.

In the 1960's a cooperative effort between industry groups produced a first attempt at these common data formats. The formats, however, were only for purchasing, transportation, and finance data, and were used primarily for intra-industry transactions. It was not until the late 1970's that work began for national Electronic Data Interchange (EDI) standards. Both users and vendors input their requirements to create a set of standard data formats that:

- were hardware independent;
- were unambiguous, such that they could be used by all trading partners;

- reduced the labor-intensive tasks of exchanging data (e.g., data re-entry); and
- allowed the sender of the data to control the exchange, including knowing if and when the recipient received the transaction.

Although today there are many syntaxes for EDI, only two are widely recognized: X12 and the Electronic Data Interchange For Administration, Commerce, and Transport (EDIFACT). These two families of standards are mandated for use within the Federal Government (FIPS 161-1)[1].

2.3 X12 and EDIFACT

In 1979, the American National Standards Institute (ANSI) chartered the Accredited Standards Committee (ASC) X12 to create a set of standards to facilitate the electronic exchange of business information. These standards define the data formats and encoding rules required for a multitude of business transactions, including order placement and processing, shipping and receiving, invoicing, payment, and many others. A sampling of common business functions with example X12 transaction sets is provided in table 1.

The work of the ASC X12 is conducted primarily by a series of subcommittees whose major function is to develop new and maintain existing X12 standards. The development and maintenance work performed by the X12 subcommittees is submitted to ANSI for national review approximately every 3 years. After successful review, ANSI publishes the X12 standards.

Due to the lengthy review and publication process, the Data Interchange Standards Association (DISA), the ASC X12 Secretariat, publishes the entire set of X12 standards annually in a document called an X12 release. The release includes the latest ANSI approved standards and new draft standards approved by the ASC X12 during that year. These releases are identified by a six digit code. The first three digits represent the version number (assigned by ANSI). The fourth and fifth digits represent the release number, and the last digit represents the subrelease number.[2] For example, X12 Version 3, Release 5 is represented by the code 003050 (pronounced *thirty fifty*).

The X12 standards were developed independent of other EDI initiatives. For example, as the first X12 standards were being deployed in North America, the European community was implementing its EDI standard: the Guidelines on Trade Data Interchange (GTDI). In an effort to create a single international EDI syntax, the United Nations Economic Commission for Europe (UN/ECE) Working Party on Facilitation of International Trade Procedures (WP.4) drew concepts from both X12 and GTDI to create the UN/EDIFACT family of standards. The EDIFACT syntax was adopted by the International Organization for Standardization (ISO) in 1987.

[2] The ASC X12 committee meets three times per year: February, June, and October. A subrelease can follow an X12 committee meeting, and would contain any draft standards approved during that meeting.

Table 1: Sampling of Transaction Sets

Business Function	Example of Transaction Set
Procurement	Purchase Order - used to provide cost and pricing data for goods and services.
Health Care	Health Care and Disability Insurance Claim - used to transmit health care service data from providers, hospitals and physician to payees.
Education	Request for Student Educational Record (Transcript) - used to request a student educational record from an educational institution that the student has attended or is currently attending.
Government Reporting	Economic Census Report - used to respond to a request from the U.S. Bureau of Census for economic census data.
Taxes	Electronic Filing of Tax Returns - will allow the sender to file tax returns with federal, state or local taxing authorities.
Warranties	Warranty Registration - used for original warranty registration, extended warranties, and service contracts.
Performance Feedback	Automotive Inspection Detail - used by automotive manufacturers, inspection agencies or carriers to report motor vehicle inspection results to other interested parties.
General Transportation	Shipment Information - used to transmit detailed bill of lading, rating, and/or scheduling information pertinent to a shipment.
Ocean Transportation	Reservation (Booking Request) - used by a shipper or forwarder to reserve space, containers and equipment for transport by ocean vessel.
Motor Transportation	Motor Carrier Shipment Information - used to transmit motor carrier bill of lading information to a motor carrier or third party.
Air Transportation	Air Shipment Information - provides the sender with the capability to transmit detailed bill of lading and rating information pertinent to an air carrier shipment.
Rail Transportation	Train Sheet - allows railroads to exchange train sheet information so that crews operating equipment on other railroads are aware of current operating conditions.
Warehousing	Warehouse Shipping Order - used by a depositor to advise a warehouse to make a shipment.
Product Development	Specifications/Technical Information - complete or partial specifications, or technical information related to products and services.

The X12 and EDIFACT standards perform equivalent functions (e.g., both define a purchase order). Although X12 is more mature and provides functions not present in EDIFACT (e.g., acknowledgments, security), many of these functions are under development in the EDIFACT organization.

The differences between X12 and EDIFACT reside in their underlying structures (e.g., the data elements and data segments - see section 2.5 for definitions of these terms). There is not a one-to-one correspondence between the X12 and EDIFACT data elements, and often, multiple X12 data elements are needed to represent one EDIFACT data element.

The differences in the X12 and EDIFACT syntaxes make interoperation impractical, if not impossible. Rather than attempt technical harmonization between the standards, the ASC X12 has adopted the EDIFACT syntax and standards. This means that the ASC X12 will develop standards based on the EDIFACT syntax.

Many X12 users view switching standards as a cost with little financial return. To appease these users, the ASC X12 committee voted (as of February, 1995) to continue the development of X12 standards (i.e., develop both EDIFACT and X12 standards), and to bring this issue to vote every 3 years. Thus, the vote to continue the development of the X12 standards will be taken again in 1998, and so forth.

2.4 EDI Implementation Conventions

The X12 and EDIFACT standards provide a great deal of flexibility regarding how application data is represented by EDI data. Some application data can be mapped to one of several different EDI structures. For example, within the X12 843 (Response to Request for Quotation) a supplier can provide pricing information in the header of the document or as individual line item entries. To remove some of the ambiguities of the standards and to ensure the successful exchange of information, trading partners adhere to *EDI Implementation Conventions.*

Implementation conventions reflect interpretations and common practices regarding the *usage* of EDI standards. They are developed and published by specific industries to facilitate the implementation of selected standards within that industry. The implementation conventions are typically updated as the standards are updated.

To provide a *single face to industry*, the Federal Government has developed a set of X12 implementation conventions to be used when transacting business with the government. By using these conventions, a federal supplier is guaranteed that all Federal agencies will interpret an EDI transaction (e.g., a bid) in the same manner. The National Institute of Standards and Technology (NIST) is the administrator for these conventions. NIST has implemented a Federal registry containing both stable conventions, and those conventions open for public comment. The documents are available in electronic format from the World Wide Web (WWW) at:

http://snad.ncsl.nist.gov/dartg/edi/fededi.html

and from anonymous FTP at:

snad.ncsl.nist.gov/pub/fededi/PFWG

2.5 EDI Standards

The X12 (and EDIFACT) standards can be viewed as hierarchical in structure. Simple data structures at the bottom of the hierarchy are combined to form more complex data structures. At the top of the hierarchy are the information units exchanged between trading partners.

This section overviews the hierarchy of X12 standards, beginning with the simple data elements, and increasing in complexity until the EDI interchange is formed. Although the EDIFACT standards use different data structures (i.e., there is not a one-to-one mapping between X12 and EDIFACT data elements) the hierarchical view is the same.

At the foundation of the X12 standards is the simple data element dictionary. A simple data element represents the smallest named item in the X12 standards. It can identify a qualifier, a value, or a description. Examples of simple data elements include:

- an invoice date,
- a weight capacity,
- an exchange rate,
- a hazardous material classification, or
- a unit of measurement code (e.g., pounds, dozens, cubic feet, gallons, etc.).

With version 3 of the X12 standards a new type of data element was defined: a composite data element. A composite data element is a set of simple data elements that represents a single named item. For example, the composite data element *C002*, which is used to identify the actions to be performed on a piece of paperwork, comprises five simple data elements. Each simple data element is a code identifying one required action. Thus, the *C002* data element specifies the set of required actions to be performed on a piece of paperwork.

Data elements are grouped into functionally related units called data segments. An example of a data segment is a geographic location, which includes the data elements for a city name, a state or province code, a postal code, and a country code. Data segments are defined in the X12 segment directory, which lists the data elements comprising each data segment in their specified order.

Data segments are grouped into transaction sets. A transaction set is the smallest meaningful set of information exchanged between trading partners. It represents a common business document, such as a purchase order or invoice. Most transaction sets are divided into three areas (called tables) which generally relate to the format of a printed document. Table 1 is the heading area, in which information pertinent to the entire transaction is placed. Table 2 is the detail area, which is usually one large loop. For example, the line items for a purchase order are placed in the detail area. Table 3 is the summary area, which provides information such as the number of data segments used in the transaction set.

Although the transaction set represents a printed document, it is not the information unit exchanged in an EDI transaction. Similar transaction sets are arranged into functional groups. For example, if *Company A* sends *Company B* two Requests for Quotations (RFQs) and five Purchase Orders (POs), the two RFQs are combined into one functional group, and the five POs are combined into a second functional group. All functional groups destined for a particular trading partner are then combined into an information unit called an EDI interchange. It is these interchanges that are transmitted between trading partners.

The functional groups of an interchange are wrapped within an interchange header and trailer. The header provides various control information, such as identifying:

- the sender and receiver of the interchange,
- the date and time of the interchange,
- the standard and version of the interchange,
- authorization and security information, and
- a unique control number by which the interchange can be tracked.

The trailer ends the interchange, providing a count of the number of functional groups in the interchange, and repeating the unique interchange control number identified in the header.

The format of an X12 interchange is depicted in figure 1. Note that interchanges are not structured to be human-readable. They contain all the information needed for their assembly and disassembly by an EDI translator, in addition to the business information being transmitted.

2.6 EDI Translation

The software component that governs the conversion of application data to and from EDI interchanges is called an EDI translator. Most EDI translators provide two services: data mapping and standards formatting. These services are shown in figure 2.

Standards formatting is the primary role of an EDI translator. To format data for an application, a translator must:

- know how to access the data, and
- understand the format of the data.

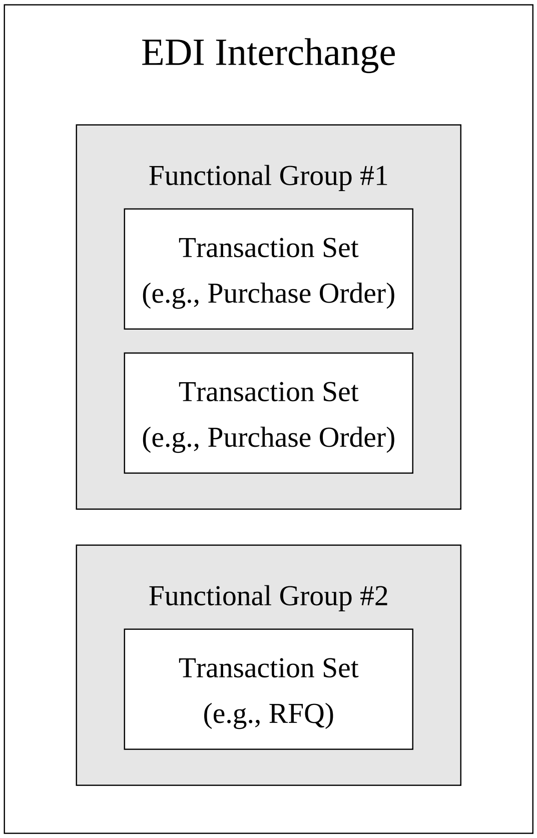

Figure 1: X12 Interchange Structure.

To access the data most translators support a file interface. For outbound transactions an application writes the transaction data to a file (called a *flat-file*). The translator formats the data according to the appropriate EDI syntax rules and produces an EDI file which is ready to be communicated to a trading partner. For inbound transactions the translator verifies that the standard version and release are supported, and that the syntax of the interchange is in compliance with the standards. The translator produces a flat-file for the application as output.

To convert flat-file data to and from EDI data, a translator must understand the format of the flat-file data. This understanding is achieved in one of two ways. First, the translator might require the user to generate the flat-file according to a format defined by the translator. This means that the user must modify the application data so it can be processed by the translator. Second, the translator might provide a tool that allows the user to specify the format of the flat-file. This tool is called a data mapper. Data mapping reduces or eliminates the programming required to integrate the translator with a business application.

Data mapping is beneficial if an application uses files for input and output. If, however, an application reads and writes data to a database rather than to files, the user needs to develop software that generates the flat-file from information stored in the database, and vice-versa. Responding to this requirement, some commercial translators now offer the capability of exchanging data directly with an application database. This removes the need for any interface software between the application and the translator.

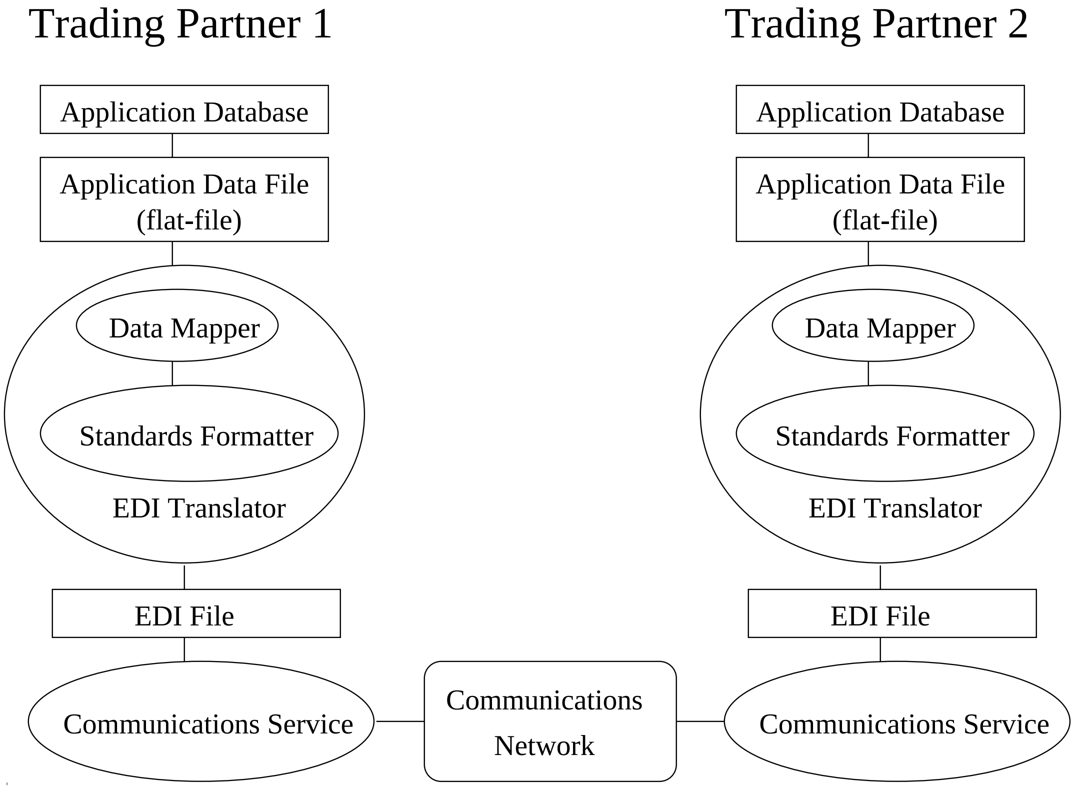

Figure 2: EDI Translation and Communications Service.

2.7 Communications

The EDI translation process converts application data to and from communications-ready EDI data. The communications service, however, is not part of the translation process. The EDI standards do not specify how EDI data is to be transmitted to a trading partner. Currently, bulk file transfer protocols (e.g., bisynchronous and asynchronous) are used to convey the majority of EDI traffic.

EDI trading partners seldom communicate directly, but rather, use the services of a third-party Value-Added Network (VAN). EDI VANs provide a communications network to connect trading partners, regardless of individual hardware platforms or communications protocols. Each partner connects to the VAN, and the VAN manages the connections to all the trading partners.

VANs also serve as a document clearinghouse, either providing a mailbox service to store received interchanges until a user is ready to download them, or proactively delivering interchanges to a user. The proactive delivery service can be immediate (i.e., interchanges are delivered as soon as they are received) or scheduled (i.e., interchanges are delivered at a specific time of day). Additionally, the proactive delivery service can be defined by document type or trading partner. For example, a user can request immediate delivery of all purchase orders, or request delivery of all interchanges from the *XYZ Company* at 3:00 PM.

2.8 Functional Acknowledgments

Although EDI standards do not specify communications information, it is of vital importance for the sender of EDI data to know if the recipient received the information. To address this concern, the ASC X12 developed a transaction set called a *functional acknowledgment* (i.e., the X12 997). The functional acknowledgment is sent by the recipient of an EDI transaction to the sender to verify the acceptance or rejection of a functional group or transaction set, and to report any syntactical errors. Many translators can be configured to automatically return functional acknowledgments when EDI interchanges are received.

3 Functional Evaluation Guidelines

This chapter builds upon the fundamental concepts presented in the tutorial by providing a comprehensive list of EDI software functions. The functions are compiled into ten categories. Each category introduces the types of functions it contains, then describes each function in detail.

This chapter is an aid for readers intending to procure EDI software. At minimum, the information provided can broaden the reader's understanding of the functionality potentially present in any EDI product. At a more comprehensive level, readers can prioritize functions to compare how different EDI products meet their requirements. Exactly how readers formulate their requirements is outside the scope of this document.

3.1 Basic Capabilities

Certain EDI capabilities might be considered mandatory by a user. This section attempts to highlight basic EDI functions that, if not available in a product, would cause a user to disregard the product.

3.1.1 Standard and Version Support

EDI users often exchange business information with many trading partners. A diverse set of trading partners may require EDI software that supports a diverse set of EDI standards. For example, a user transacting business with four trading partners might need to support two versions of X12 and two versions of EDIFACT. In addition, some trading partners might mandate more specialized EDI syntaxes, such as the Warehouse Industry Network Standard (WINS). When purchasing EDI products, users must ensure that the products support all the EDI standards and versions used by their trading partners (if known). Most commercial translators support, at a minimum, the current version of X12, and its two previous releases.

3.1.2 Document Support

To transact business with multiple trading partners, some users will need to handle many types of business documents, such as revenue receipts statements, court notices, tax rate notifications, and others. Although most EDI software packages support an entire set of EDI standards (e.g., all X12 transaction sets), some business applications that include an EDI component might handle only specific subsets of the standards. An EDI-enabled finance application, for example, might support finance documents only. A user must confirm that the EDI software supports all required documents types.

3.1.3 Operating System Compatibility

A user might require an EDI product to run on a specific platform, such as Disk Operating System (DOS). Some EDI products are available on many platforms, ranging from PCs to mainframes. Other products can only be used on one platform.

Operating system compatibility becomes a more complex issue when a time-sharing system (e.g., Microsoft Windows) is used overtop a sequential system (e.g., DOS). Due to event scheduling and timing issues, it might be difficult to seamlessly integrate a Microsoft Windows business application with a DOS-based translator.

One last issue regarding operating system compatibility is the creation of EDI maps (i.e., the output of an EDI mapping utility) that can run on multiple platforms. This function might be used by a large company with multiple EDI-enabled sites. At one site the company deploys a fully functional EDI system which develops EDI maps for the other sites. Thus, the other sites only need to deploy and maintain a limited EDI system (e.g., one that performs translation, but does not perform data mapping). Since these other sites might use different hardware platforms, purchasing an EDI product that produces EDI maps for a variety of platforms can save time and expense.

3.1.4 Binary Data

Most EDI exchanges involve the transmission of text data. If, however, a user exchanges specifications or technical information with a trading partner (e.g., the X12 841 - Specifications/Technical Information), the user's EDI software must support the exchange of binary data.

3.2 Access Control

An EDI system typically has many users. An insurance office, for example, might have ten claims examiners served by one EDI product. In such a multi-user environment, controlling user access is important. This category describes how an EDI product might implement access control.

3.2.1 Login Procedure

One method of restricting access to EDI software is by implementing a login procedure. Only users with valid login names and passwords are permitted access. This restriction provides minimal security against illicit users.

3.2.2 Selective Accessibility

EDI users can be viewed as belonging to one of two categories: technical users and non-technical users. Technical users require access to administrative components, such as those needed to configure communications data, trading partner data, and mapping data. Non-technical users only need to perform routine operations, such as starting the software. Selective accessibility allows users to be divided into groups such that a particular group of users is granted or denied specific access privileges.

3.2.3 Concurrent Activity Control

In a multi-user environment disallowing some concurrent activities is desirable. These disallowed activities center around system administration functions, such as modifying trading partner network information while a communications session is in progress, or allowing two users to update mapping information simultaneously.

3.3 Communications

When implementing an EDI system, some type of data communications software is necessary. Communications software can perform a variety of services, such as transmitting EDI files to VANs or encapsulating EDI data in electronic messages. Functions pertaining to the communication of EDI data are described in this category.

3.3.1 Communications Software

Many EDI products include some type of file transfer software. This software initializes a modem to call a VAN and to upload and download files. If an organization lacks communications capabilities, purchasing a communications-enabled EDI translator is a viable option. Another option is to integrate a translator with an external communications system. See section 5.2 for more information about integrating translators with external communications systems.

3.3.2 Protocol(s) Support

A user purchasing a communications-enabled translator must ensure that the communications software supports the required protocol or protocols. Many EDI packages, especially for PCs, include asynchronous communications software. Some EDI packages provide bisynchronous communications software, but few are integrated with electronic messaging software (e.g., Simple Mail Transfer Protocol (SMTP) or X.435).

3.3.3 VAN Script Files

Traditionally, the communication of EDI data has involved third-party VANs. EDI VANs provide many services for their subscribers, including electronic mailboxing of EDI transmissions, protocol conversions, and EDI audit trails.

When communicating with a VAN, a subscriber initiates a session (i.e., a dialogue between the subscriber's computer system and the VAN's computer system). The session is governed by a predefined set of commands called a *VAN script*. The VAN script coordinates specific actions, such as:

1. dialing into the VAN's network service,

2. identifying the login name and password of the subscriber,

3. downloading any EDI interchanges waiting in the subscriber's mailbox,

4. uploading any EDI interchanges to be delivered to the subscriber's trading partners.

There is no standard set of commands for communicating with VANs; a script that operates with one VAN might not operate with a different VAN. When purchasing a communications-enabled translator, a user should determine whether the software includes a script for communicating with the desired VAN. If the user has not yet selected a VAN, EDI products that provide scripts for all major VANs are available.

Some VANs provide the subscriber with the software required to interact with their computer system. In this case, the VAN script performs a simplified function, such as transferring a file to a specific file directory and starting the VAN's proprietary communications software.

3.3.4 Multiple VAN Support

An EDI user might conduct business with trading partners that subscribe to different VANs that are not interconnected. These users must ensure that the communications software is capable of connecting to multiple VANs.

3.3.5 Direct Trading Partner Connection

Some trading partners might not use VAN services, requiring the user to connect directly to their computer systems. To accommodate these trading partners, the user's EDI system must support direct connections.

Direct connections can place additional requirements on the receiving communications software. If a user is to receive EDI interchanges directly from a trading partner, the user's communication system must operate in a mode where it waits to receive a transmission. For a single-user system, such as DOS, this means dedicating one PC to receiving modem calls.

3.3.6 Script Building Tool

A user might be interested in communicating with a VAN for which no script is available, or connecting directly to a mainframe computer. For these scenarios an EDI product might provide a script building tool which helps the user create custom scripts for connecting to VANs or mainframes.

3.3.7 Unattended Transmissions

If an EDI product supports VAN scripts, the product might allow the user to schedule when the scripts are executed. For example, a user can configure the communications software to call a VAN twice per day, except on weekends and specific holidays.

3.3.8 Manual Transmissions

Although the scheduled transmission of EDI data is a desirable function, permitting a user to manually start the communications process is also useful. Manual control of the communications software facilitates its initial configuration and aids with correcting communication errors.

3.3.9 Communications Audit Trail

A communications audit trail provides the user with a log detailing the transmission of each interchange. Information typically provided with an audit trail includes: times, dates, identifiers, acknowledgments, errors encountered, etc. Audit trails are useful for debugging transmission problems, generating reports, and verifying that an interchange was sent or received by a trading parter.

3.3.10 Viewing Utility

There is a variety of communication data which might need to be viewed by an EDI user. This information includes scheduled transmissions, audit trails, outstanding functional acknowledgments, configuration data, and others. Rather than manually editing files, an EDI product might provide a utility for viewing various aspects of communications data.

3.4 Installation and Maintenance

EDI software is often shipped on a portable medium, such as diskette or tape, and needs to be installed on a physical hard drive. After the software is installed, it must be managed and maintained. This category describes functions pertaining to the installation and maintenance of EDI software.

3.4.1 Automated Installation Routines

Automated installation routines remove many of the complexities of installation from the user. Most routines verify required resources (e.g., sufficient disk space), create required file directories, modify relevant system files, and copy the appropriate data from the installation medium to the destination file directories. These routines typically confirm the creation of file directories and make copies of system files prior to changing them.

If an EDI product does not provide installation routines, more effort is required on the part of a user. The user might need to manually create file directories, edit system files, and copy or extract files from the installation medium.

An EDI product might provide different routines for new installations, upgrades, and mode changes. A new installation refers to installing EDI software on a system for the first time. An upgrade refers to software which augments existing EDI software. A mode change refers to the capability of changing execution modes (e.g., changing from *demonstration mode* to *production mode*). This capability gives the user the flexibility to install the necessary software without re-keying information.

3.4.2 Updating Standards and Versions

EDI standards are updated periodically. In order to make full use of functionality provided by the latest standards, the updates need to be incorporated into the EDI software. If the EDI software is table-driven (i.e., indexed files are used to maintain the EDI standards), updating the standards is accomplished simply by updating the relevant tables; no reprogramming is required. If the EDI software is not table-driven, updating the EDI standards can be a more cumbersome process.

EDI vendors provide updates either automatically (e.g., within 30 days of a new release of the standard) or upon user request. The updates can be distributed on a diskette or tape, downloaded to the user's EDI system, or placed on a bulletin board. Users should inquire as to the timeliness and method by which updates are supplied.

3.4.3 Tracing

Tracing provides a user with a detailed description of how EDI data is processed. For example, a tracing utility can log the mapping of every application data element to its EDI counterpart. Tracing is used mainly on an ad hoc basis, such as for initially testing the EDI software and correcting translation errors.

3.4.4 Logging

Logging is the writing of transaction information to short-term storage (e.g., the main disk for the EDI system). Logging provides a transaction audit trail for the EDI user.

An EDI product might allow the user to specify a log level. For example, a user might request to log the entire EDI header for each interchange, or might request only to log dates, unique EDI identifiers, and trading partner identifiers.

3.4.5 Archiving

Archiving is the writing of transaction information to long-term storage (e.g., magnetic tape). Archiving provides permanent records for the EDI user.

Archiving can be performed manually by the EDI user or automatically by the EDI software. If performed automatically, the user specifies the conditions by which transactions are archived (e.g., archive all transactions older than 1 year). Some EDI software compresses the data being archived to conserve space.

3.4.6 Automated Purging

EDI data which has been logged or archived might need to be purged. Some EDI products allow a user to specify criteria against which purging is performing automatically. These criteria include: trading partner identifiers, document type (e.g., transaction set) identifiers, and starting and ending dates.

3.4.7 Data Recovery and Restart

Data recovery and restart enable an EDI system to automatically recover from a power or communications failure. If an EDI system, for example, experiences a power failure while transmitting data to a VAN, the EDI system retransmits the data when power is restored.

3.5 Application System Interfaces

A user has several options for implementing an EDI system. One option is to purchase an EDI-enabled business application. Another option is to integrate EDI software with an in-house application. If the user chooses this second option, there are several methods by which the integration can be accomplished. These methods are detailed in this category.

3.5.1 File Interface

Most translators interface with applications by exchanging files. For outbound transactions an application writes data to a file which is read and processed by the translator. For inbound transactions the translator writes data to a file which is read and processed by the application.

If the EDI software contains a mapping utility, the user can specify the format of the files exchanged with the translator. If no mapping utility is present, the format of the application files might need to be modified to accommodate the translator.

The flexibility of the mapping utility is a critical issue. In an ideal scenario a business application's import/export utility creates a data format acceptable to the EDI mapper. More commonly, however, the format of the application data cannot be supported by the mapper, and the user must develop (or have developed) software that reformats the application data. Data reformating increases implementation time (e.g., complex reformating software might take weeks to develop) and degrades performance (e.g., complex reformating software can double or triple the time required to convert application data to and from EDI data).

3.5.2 Database Interface

Some business applications do not use data files, but read and write information directly to a database. To interface with these applications, some translators can be configured to generate the appropriate database commands for reading and writing information to the application database.

3.5.3 Printing

Although EDI minimizes the need for human intervention, a user might require paper copies of documents. If a company conducts business with both EDI and non-EDI partners, some translators can create EDI interchanges for the EDI partners and paper forms for the non-EDI partners.

Paper copies might also be required for inbound transactions. Assume a company's purchasing and shipping departments do not communicate electronically. A translator might receive purchase orders electronically (e.g., X12 850s), create flat-files for the purchasing application, and create paper forms containing the purchase order data to be sent to the shipping department.

The format of the data being printed is often user-defined (i.e., the user specifies how the data being printed is to appear).

3.5.4 Forms

Some EDI products act as stand-alone applications. This means the software provides on-line forms for viewing EDI data. These forms are presented in a user-friendly format, and require no knowledge of EDI standards.

Electronic forms are useful for organizations that do not have an application to generate or receive EDI data. For example, a small supplier wants to receive Federal Requests For Quotations (RFQs). This supplier can purchase a stand-alone product that will not only accept an EDI RFQ, but display the RFQ data on the screen.

3.5.5 Related Document Generation

Some EDI products allow a user to create documents from data contained in other documents. In the above example, the software purchased by the small supplier may not only display the RFQ data on screen, but allow the supplier to enter price data. Once the price data is entered the software automatically generates an EDI bid based on the RFQ data and the price data. Related document generation saves time and minimizes typographical errors.

3.5.6 User-Defined Forms

The forms package provided by an EDI product might allow a user to define how data appear on the screen, as well as special handling functions for individual fields. These functions include:

- the line and column number for where a field is to be displayed,
- the length constraint for a field,
- automatically computed fields (e.g., line item totals in a purchase order),
- minimum/maximum value checking (e.g., a maximum discount of 25% can be specified for a field),
- automatic capitalization (e.g., state abbreviations),
- validated fields (e.g., validate a state abbreviation against a table of state abbreviations),

- mandatory field checking (e.g., purchase order number must be present),
- automatic dates (e.g., insert today's date on a purchase order),
- automatic counters (e.g., line item number on a purchase order), and
- simple graphics (e.g., for lines and boxes).

In addition, a forms package might allow a user to enter data by selecting items from a list. For example, rather than typing a buyer's name, the user selects the buyer from a list of buyers. List selections can be based on system defaults or by values previously entered by the user.

3.5.7 Code Expansion

Some forms packages convert EDI codes to descriptive text. For example, the line item section of a purchase order can contain codes to describe a product (e.g., CB for buyer's catalog number, PA for pattern number, BO for buyer's color). The EDI software expands these codes to descriptive text (e.g., Catalog Number, Pattern, Color), when displaying the information to the user.

3.6 Customization

Customization functions enable an EDI user to tailor an EDI system. Specific information, such as trading partner profiles and error handling, can be configured into an EDI product. This category describes the functions that enable EDI products to meet the specific requirements of an organization.

3.6.1 User-Defined Configuration

Some EDI products are configured to a user's needs by the EDI vendor. For example, if a user requires seven purchasing-related documents, the EDI vendor configures them for the user. If the user, at some later time, requires another document to be configured, the user must contact the vendor. The vendor adds the new document, and brings a new version of the software to the user.

Other EDI products are configurable by the user. This means the user can incorporate modifications (e.g., map new document types or add new trading partners), into the software. This is a desirable function if the user's needs are subject to growth and change.

3.6.2 Standards Referencing Tool

To assist with data mapping functions (e.g., adding a new document type) an EDI product can provide on-line tables of EDI standards. These tables identify the version and release of a standard document (e.g., version 3010 of the X12 840), and all its data segments and data elements.

3.6.3 Trading Partner Profile

Trading partner profiles enable a translator to accommodate the requirements of multiple trading partners. They contain the information needed to format outbound interchanges for a specific trading partner, and to recognize interchanges received from a specific trading partner. This information includes:

EDI Standard, Version, and Release: Identifies the version and release of the EDI standard used to format interchanges.

Transaction Identifiers: Identifies the documents that can be exchanged (e.g., X12 810).

Data Element Delimiter: Identifies the character (e.g., the ASCII characters "+" or "*") used to delineate the data elements contained in an EDI document.

Segment Terminator: Identifies the character (e.g., the ASCII characters "," or ".") used to delineate the data segments contained in an EDI document.

Exchange Direction: Identifies the direction of the data exchanged (e.g., inbound, outbound, or both).

Transaction Segments: Identifies all the EDI data segments that are permitted for a document.

Network Identification: Identifies network and communications information, such as network addresses and telephone numbers.

3.6.4 Automatic Trading Partner Entry

If an organization participates in an open EDI environment, transactions can be received from unknown trading partners. Rather than reject a transaction because the originator is not registered, a translator can automatically create a trading partner profile entry from information contained in the inbound transaction.

3.6.5 Multiple Functional Groups

Some translators permit multiple functional groups in one interchange. For example, if three invoices and one RFQ response are being sent to the same trading partner, the EDI software creates one interchange containing two functional groups: one invoice functional group and one RFQ response functional group. If the software does not support multiple functional groups, two interchanges need to be created. The second interchange causes increased overhead, for both transmission costs and storage requirements.

3.6.6 Distinct Header Identifiers

An EDI interchange contains an envelope header (e.g., the X12 ISA) and functional group headers (e.g., the X12 GS). Both the ISA and GS headers contain identifier fields. The ISA values identify the sender and recipient of the interchange. The GS values can be used to identify different applications within an organization. For example, if an interchange contains two functional groups with two distinct GS recipient identifiers, the receiving EDI software can translate the interchange and send data from the first functional group to one application, and data from the second functional group to a different application.

If a user intends to route data based on functional group identifiers, the user must ensure that the EDI software permits distinct values in the functional group identifier fields. Some EDI software simply copies the ISA header identifiers into the GS header identifiers.

3.6.7 Document Type Sequencing

Function groups within an interchange are identified by a control number. Document types (e.g., transaction sets) within a functional group are also identified by a control number. Some EDI products assign functional group and document control numbers sequentially for each trading partner. The benefit of a such a numbering system is that a trading partner can determine if a document is missing from a transmission by viewing the document control numbers. If there is a lapse in the sequence of control numbers, a document is missing.

3.6.8 Functional Acknowledgment Generation

Most commercial EDI products have the capability to send and receive the functional acknowledgment transaction set (i.e., the X12 997). The functional acknowledgment transaction set identifies the status of a received EDI transaction.

Some EDI products can be configured to generate functional acknowledgments automatically upon receipt of an EDI interchange. There are generally three levels of acknowledgments:

Group: indicates whether a group of like documents contains errors,

Set: indicates whether each document contains errors, and

Segment/Element: indicates the exact segment and element that is in error.

3.6.9 Modify Menus

EDI products often use menus to prompt users for information. A user might be allowed to customize these menus. Customization options include: changing the wording or ordering of options, adding new options, deleting default options, and creating new menus.

3.6.10 Language Version

Some EDI products have the capability to display information (e.g., menus and prompts for information) in languages other than English.

3.6.11 Macros

A macro is a set of instructions programmed for and executed by one key on the keyboard. The benefit of macros is that they save keystrokes when performing repetitive tasks. During the mapping process, for example, a user might need to traverse several levels of menus to equate one application data element with a standard data element. Since many data elements might require mapping, allowing a user to program one key which automatically traverses all relevant menus is useful.

3.6.12 Error Handling

EDI products can handle errors differently. When an error occurs, the software should, at minimum, display or log a notification of the error. Information such as the location of the error and a meaningful description of the error is helpful. A user can reference the appropriate documentation to gain more information about the error, as well as possible solutions.

A user might be permitted to specify error handling instructions on a trading partner basis. For example, if a specific trading partner generates a specific error (such as exceeding the maximum length of a text description), the EDI product can be configured to automatically: generate a functional acknowledgment, attempt to correct the error and continue, by-pass the transaction in error and continue, or halt and notify the user with a diagnostic message.

3.6.13 New Document Types

Some EDI software enables a user to define new document types. This function is useful for testing document types being developed by EDI standards organizations, or for designing and generating very specific documents required for intra-business purposes.

3.6.14 Security Package

For some EDI users security is a critical issue. Although VANs provide limited security by requiring subscribers to authenticate themselves with a password, users might require stronger security services. To accommodate these users, some EDI products provide a security package conforming to the X12.58 standard. The X12.58 authentication and encryption services:

- verify the identity of the EDI sender to the recipient to detect impersonation,
- verify the integrity of the transaction by detecting changes,
- protect against insertion, deletion, or replay of transactions via a unique identifier, and
- ensure that only authorized users view the data.

Security services can be enabled on a per-recipient and/or per-transaction basis. For example, *User A* has two trading partners: *User X* and *User Y*. *User A* configures the EDI software, such that:

1. RFQs are not authenticated (per-transaction basis),
2. transactions to *User X* are not authenticated (per-recipient basis), and
3. outbound POs to *User Y* are authenticated (per-transaction and per-recipient basis).

3.7 Data Conversion and Editing

EDI software enables a user to transact business with many trading partners. When dealing with multiple trading partners, some degree of flexibility is required to support their different needs. This category describes functions where EDI products automatically, or when initiated by the user, modify trading partner data to ensure compliance to the standards and to facilitate integration with the user's application system.

3.7.1 Character Set Conversion

Trading partners might use different character sets (e.g., ASCII and EBCDIC). Some EDI products convert between the character set used by the sending trading partner and the character set required by the receiving trading partner. This function is pertinent only to sites that use point-to-point communications (i.e., direct trading partner connections). If a VAN is used to deliver the EDI data, the VAN performs any required character set conversions.

3.7.2 Code Conversion

Codes used by an EDI user might be different than standard EDI codes. For example, the X12 Product/Service ID Qualifier for a serial number is *SN*. A user's application might use the code *SERNUM* to identify a serial number. To faciliate integration between the user's application and the EDI software, the EDI software can convert the standard code to and from the user's code.

3.7.3 Automatic Compliance Correction

Standards compliance procedures verify different types of information, including the EDI standard and version being used, the identity of the trading partner, and the syntax of the data. To perform compliance checking, EDI software references its tables of EDI standards and the user's trading partner profiles. This process can occur for both inbound and outbound data.

Some compliance errors can be corrected by the EDI software. If the error is simple in nature (e.g., the maximum length of an inbound freeform description is exceeded), the EDI software can automatically adjust the data to make it comply with the standards (e.g., can truncate the description to the maximum permitted length).

3.7.4 Manual Compliance Correction

Some compliance verification errors are too severe for the EDI software to make an automatic adjustment. In this case the software might suspend processing so a user can review the transaction, correct the error, and submit the transaction for reprocessing. This feature is usually provided in a user-friendly format (i.e., no EDI knowledge is required to review and correct the error).

Users must be aware of the ramifications of compliance correction functions. If inbound data is modified, and the EDI transaction is digitally signed, the modification invalidates the signature. In addition, if a recipient changes an EDI transaction, the change might have legal consequences.

3.7.5 Duplicate Number Detection

Some EDI products track the use of business document numbers, such as purchase order numbers. If a duplicate number is identified, the software displays or logs an error message, or halts processing of the transaction until the user can correct the duplicate.

3.8 Control and Audit Reports

Reports provide a means for an organization to view and maintain information on various aspects of an EDI system. Reports can be viewed on the screen, printed, or archived. This category describes some of the types of reports that an EDI product might generate.

3.8.1 Error Reports

An EDI product might provide a report summarizing problems encountered by the software (e.g., data mapping errors, communications errors). Error messages are typically elaborated in product documentation with descriptive text and possible solutions.

3.8.2 Functional Acknowledgment Reconciliation

Functional acknowledgment reconciliation is a feature that matches outbound EDI transactions with received functional acknowledgments. This function enables a user to verify whether a trading partner received an EDI transaction. A report can be produced providing the user with a list of transactions that have been acknowledged, and a list of those with outstanding acknowledgments. For example, every outbound transaction can be associated with an acknowledgment status, such as:

- acknowledgment not requested;
- acknowledgment expected;
- acknowledgment arrived successfully;
- acknowledgment arrived with errors and accepted;

- acknowledgment arrived with errors and partially accepted;
- acknowledgment arrived with errors and rejected.

3.8.3 Transaction Reporting

An EDI product might provide a report summarizing EDI transactions that have been sent and received. The summary might be based on selected parameters, such as by trading partner or date.

3.8.4 Transmitted Characters

An EDI product might provide a report identifying the number of characters transmitted to various trading partners (usually defined as the number of bytes). This report is helpful for reconciling a network's transmission bill.

3.8.5 User-Defined Reports

Users might be allowed to define reports based on a wide range of criteria, such as inbound and outbound transactions, functional acknowledgment reconciliation, updates to trading partner profiles, updates to archived transactions, and others. The user configures the information, format, and frequency to which these reports are generated.

3.8.6 Report Levels

Different users might require different levels of reporting. One user might want only high-level transaction information, such as dates, interchange identifiers, and sender and recipient identifiers. A different user might want all the high-level data, plus more detailed information, including functional group identifiers, functional group control numbers, and the number of documents contained in each functional group. Report levels are generally selected by a user from a predefined set of choices configured into the EDI software.

3.9 Support

Once an EDI product is purchased, users often require support with the installation, maintenance, and use of the software. Support functions can be provided by the vendor (e.g., product training) or by the software (e.g., on-line tutorials). This category describes vendor initiatives and software functions whose purpose is to support the user.

3.9.1 Help Screens

Help screens provide users with a quick reference concerning software features. They range from brief descriptions of data fields to detailed explanations of software functions. Help screens can also provide suggestions for entering data or instruct a user to reference specific sections of paper documentation.

3.9.2 User Documentation (Paper)

User documentation provides narrative text concerning the daily use of the EDI software. Typically, such documentation is indexed into coherent sections which provide the user with more detail than help screens.

3.9.3 Technical Documentation (Paper)

Technical documentation provides information for users needing a detailed understanding of the product. Product functions relating to trading partner maintenance, data mapping, communications, report generation, functional acknowledgment reconciliation, and transaction archiving are typically included with technical documentation.

3.9.4 On-line Tutorial

An EDI product might provide an on-line tutorial. Tutorial packages guide users through the software features provided by a product. They provide formal step-by-step approaches for specific software functions (e.g., data mapping).

Tutorial packages offer a controlled environment where users can enter and manipulate test or *dummy* data. Tutorials typically display help messages describing valid values, and only permit valid input or edits. Some tutorials enter data in various fields or perform specific operations automatically for the user.

Tutorials packages provide an effective method to educate users concerning the operation of an EDI product. Users also gain familiarity with EDI standards and the EDI process as they use the on-line tutorials.

3.9.5 Vendor Services

Some vendors perform miscellaneous services for their customers, including: on-site installations, mapping of EDI transactions, writing communications scripts for EDI VANs, and others. These services are often provided at additional cost.

3.9.6 Customer Support

Nearly all EDI vendors provide customer support (e.g., a telephone number or electronic mail address). If a user cannot solve a problem given the product documentation, the user can contact a technical representative.

It is important to be aware of customer support response time. Many phone support services are handled in one of the following two ways: the user enters a response queue until the next available support person becomes available; or the user leaves a phone message and return phone number, and a support person returns the call. Some customer support phone numbers are toll-free.

3.9.7 Training

An EDI vendor might provide training about an EDI product. Details of installation, maintenance, and operation are presented in a classroom environment. Training costs might be additional to, or included in, the EDI product price. Training sessions might be held at the user's site (sometimes at an additional cost) or at the vendor's training facility. Sessions typically range from one day to one week for complete training.

3.9.8 User Groups

Some vendors organize product user groups for their customers. Product user groups are groups of customers that use the same software product. They meet to discuss issues relevant to the product, including problems, solutions, and possible future improvements.

3.10 Miscellaneous

Many of the functions in the previous categories have been concrete (i.e., the EDI product provides the function or it does not). There are some issues that should be considered when evaluating an EDI product for purchase that are less tangible or less relevant to the nature of the software than those presented above. These issues are described in this category.

3.10.1 Cost

Cost is a multifaceted issue. The cost of an EDI product can be divided into three areas:

Software: costs pertaining to the purchase of EDI software.

Hardware: costs pertaining to the purchase of hardware required for the EDI product, such as LAN cards or upgrades to computer systems.

Fees: costs pertaining to annual fees, which include maintenance, software upgrades, new releases of the EDI standards, and support.

The budget of the user determines the importance of cost as an evaluation factor.

3.10.2 Escrowed Source Code

For the protection of the user an EDI vendor can make the source code available in escrow. In the event that the vendor goes out of business and the user decides to continue using the software, the source code can be modified and maintained.

3.10.3 Effectiveness

Effectiveness pertains to the ability of the EDI software to accomplish a specific task. For example, the EDI mapping software might provide a user with many functions, but its use might not be intuitive, forcing the user to consult documentation frequently. Also, the EDI vendor might provide appropriate documentation (e.g., User's Guides, Technical Guides), but the documentation might not be well organized, or might be difficult to understand.

To better appreciate the effectiveness of an EDI product, the user has several options. The user can request a copy of the EDI documentation from the vendor. By examining documentation in advance, the user can better determine its adequacy and understandability. The user might also be able to determine how easy the product is to install, configure, debug, and use. The user can inquire as to the typical start-up time for the product. Start-up times can range from an hour to a month, depending on the product, platform, and amount of customization. Products with minimal start-up times are often less complex (and possibly less functional) than products with greater start-up times. Another option is for the user to request demonstrations of the product. A demonstration will provide an overall view of the product, especially concerning its ease of use.

3.10.4 Business Applications

Some vendors might provide business applications that are integrated with their EDI software. For example, a vendor might have a purchasing or finance application that drives the EDI software. If a user wants to upgrade an application, or does not have an in-house business application, purchasing EDI software from a vendor that also supplies business applications might be a desirable option.

3.10.5 Vendor

Users should verify the qualifications of the EDI software vendor. Several issues pertaining to the vendor include:

- How long has the vendor been in business?
- How many sites are using the vendor's EDI software?
- Is the vendor that is marketing the product also its developer?

A user can request references from the vendor to determine if the vendor is fulfilling the requirements of other customers.

4 Performance Evaluation Guidelines

The previous chapter outlined software capabilities to assist a user with evaluating the functionality provided by an EDI product. This chapter recommends a procedure for evaluating the performance of the translation component of that EDI product. The chapter is divided into two sections. Section 4.1 describes EDI translator testing scenarios, and section 4.2 provides a performance testing example.

Time and expense are involved in evaluating performance. Prior to initiating an evaluation, a user should first determine if an evaluation is required. The following two examples can assist a user with making that determination.

Example 1: Some businesses process a relatively small number of transactions that are distributed evenly throughout the day. For example, a purchasing office might process 100 procurement-related transactions per day. Since 100 transactions processed over an 8-hour work day averages to less than 13 transactions per hour, and any commercial EDI translator should be capable of performing acceptably under these conditions, this user would most likely **not** be concerned with performance.

Example 2: Some businesses process large numbers of transactions per day, or experience periodic *spikes*. For example, an insurance company might process hundreds of health care claims per day, or the Internal Revenue Service (IRS) might receive thousands of tax returns filed electronically on April 15. These users would most likely view performance as a critical requirement.

Once a user decides a performance evaluation is required, the user should contact the software vendors for the EDI translators under evaluation, and request them to perform the testing. The vendors, rather than the user, should obtain the performance measurements for the following four reasons.

1. The vendor should have the hardware and software required to run the EDI translator. Thus, the user avoids procuring or leasing hardware and software to perform the testing.

2. The vendor should have the expertise to install the translator on a test system, or it might already be installed. Thus, the user avoids having to install and configure the translator.

3. The vendor should have access to the source code so as to make any modifications needed to perform the testing (e.g., displaying start and stop times). Thus, the user avoids the problems associated with acquiring and modifying source code.

4. The vendor should have the expertise to optimize the performance of the translator. It is important when comparing the performance of different translators to compare the best performance of each one, so the comparison is fair. Thus, the user avoids learning how to optimize each translator being tested.

To obtain fair measurements the performance tests for different translators must be performed on similar platforms. For example, if testing DOS-based translators, the following hardware characteristics must be the same:

- the hardware model (e.g., 486 DX),
- the processor speed (e.g., 50 MHz),
- the amount of Random Access Memory (e.g., 4 MB), and
- the hard drive speed (e.g., 18 ns).

In a multi-user system, additional issues, such as percentage utilization of the Central Processing Unit (CPU) by the translator, need to be considered.

4.1 Performance Tests

Once the user establishes the hardware platform, the user must instruct the vendors on how to perform the testing. The remainder of this section provides recommendations for defining a performance measurement. The recommendations assume the user has a business application with which the translator must be integrated, and that there is a flat-file interface between the translator and the application. Note that the recommendations measure the EDI translation process only (i.e., no communications functions are measured).

First, the user should obtain a copy of the data files for typical transactions processed by the application. For example, if the application is used for small purchasing within the Federal Government, the user can obtain a copy of a typical Request For Quotation (RFQ) data file and a typical Purchase Order (PO) data file. If the user has identified an EDI trading partner, the user can also obtain a copy of a typical EDI transaction generated by the trading partner. In the above example, the user can obtain a copy of an X12 843 (i.e., the trading partner's response to an RFQ).

The data files must then be shown to the vendors, and the EDI implementation conventions to be used (e.g., the Federal EDI Implementation Conventions) must be identified. There is little chance that the application data files can be mapped to their EDI counterparts without modification. The user and vendors should discuss possible changes to the format of the application data files until the user is comfortable with the changes. Although each vendor might deviate slightly from the agreed format, the format should be similar enough between all the vendors to ensure fairness with the performance testing.

At this point the user has identified a set of outbound data files (e.g., RFQs and POs) that will be translated into EDI interchanges, and a set of inbound EDI interchanges (e.g., RFQ responses) that will be translated into application data files. The user can now choose between one of two testing scenarios. These scenarios are described below.

4.1.1 Transaction-Based Testing

With transaction-based testing the user specifies a number of transactions to be translated. For example, the user might request that 100 RFQs, 100 RFQ responses, and 100 POs all be translated to determine the performance measurement for a translator. In this case the vendor would:

- Record the start time for the test.
- Perform the translation for the 300 transactions.
- Record the stop time for the test.

The user can simply divide the difference between the stop and start times by the number of transactions to determine the average time to translate one typical transaction.

4.1.2 Time-Based Testing

With time-based testing the user specifies a time period (e.g., 15 minutes) during which as many transactions as possible are translated. The test should be run two times: once for outbound transactions and once for inbound transactions. This is because it might not be possible for a translator to alternate between inbound and outbound transaction processing. In this case the vendor would:

- Start the outbound test.
- Perform as many outbound translations as possible.
- Stop the outbound test.
- Start the inbound test.
- Perform as many inbound translations as possible.
- Stop the inbound test.

The user is provided with the maximum number of typical outbound and inbound transactions that can be translated during a given time period.

4.2 Performance Testing Example

To facilitate the understanding of EDI translator performance testing, a performance testing example is provided. The example follows the recommendations put forth in section 4.1. These recommendations are summarized below.

1. Compose a list of EDI translators to be evaluated.
2. Contact the EDI software vendors to perform the evaluation.
3. Establish the hardware platform.
4. Collect application data files.
5. Select the appropriate EDI implementation conventions.
6. Specify the test scenario.

For this example, the evaluation is performed using four commercial EDI translators installed in the NIST Electronic Commerce Integration Facility (ECIF) procurement testbed (see appendix A). They represent four translators that a user wishes to evaluate.

Although the recommendations state the testing should be performed by the EDI software vendors, NIST personnel conduct the testing for this example. This change is acceptable for two reasons. First, NIST personnel have installed and configured the translators in the testbed, and have the expertise required to conduct the tests. Second, the goal of this testing is *not* to determine which translator performs best, but rather to determine if there is a significant variance in the performance of the EDI translators installed in the testbed. Since only *ballpark* measurements are desired, no modifications to source code are needed.

All the DOS-based translators in the testbed are installed on 66 MHz 486 DX machines. This machine represents the user's hardware platform.

The recommendations state the user must obtain copies of application data files. Since no application is integrated with all four translators, an alternative approach is needed. This approach uses the Federal EDI Implementation Conventions.

The Federal EDI Implementation Conventions contain example X12 transaction sets. For example, the 840 transaction set (i.e., the RFQ) includes three examples: a simple RFQ, a simple RFQ for services, and a complex multiline RFQ. In lieu of application data, information in the simple RFQ and complex multiline RFQ examples is used to represent application data. This approach provides both valid RFQ data and the flexibility to format the data so it is acceptable to the translators in the testbed with only minor variations. The RFQ test data is included as appendix C.

The final step is to select the testing scenario. For this example, the transaction-based scenario will be used to translate 50 simple RFQs and 50 complex multiline RFQs.

4.2.1 Test Procedures and Results

The four translators being tested are referenced as *A, B, C,* and *D.* For each translator, the testing is conducted as follows:

1. One input file is used. This file contains the 50 simple RFQs and the 50 complex multiline RFQs arranged in alternating order.

2. The execution of the test is controlled by a DOS batch file. The batch file displays the current time (i.e., start time), invokes the translator to perform the translation, displays the current time (i.e., end time), then exits.

3. One output file is created. This file contains the one hundred EDI interchanges resulting from the translation of the 50 simple and 50 complex multiline RFQs.

4. The test is run three times.

The results of the performance tests are displayed in the following table. For each translator the difference between the start and end times for each of the three runs is shown. The average of the three runs is also shown.

Translator	Run 1 (sec)	Run 2 (sec)	Run 3 (sec)	Average (sec)
A	23.79	22.02	21.86	22.56
B	17.86	17.03	17.09	17.33
C	19.87	19.44	20.16	19.82
D	21.53	20.91	21.12	21.19

4.2.2 Test Observations and Analysis

The fastest and slowest translators are *B* and *A* respectively. On average, translator *B* processed the 100 RFQs 5.23 seconds faster than translator *A*. The percentage difference between the two translators is 23.18%.

There are no visible differences between the processing of the four translators. For example, no intermediate files are created to temporarily store data being translated. Thus, the most likely explanation for the difference in translation times is the internal algorithms used to perform the translations.

From a percentage standpoint, translator *B* performed much better than translator *A* (23.18%). The significance of this measurement to an organization depends on the number of transactions it processes per day, as well as its performance requirements. Although 23.18% seems high, the time difference for the hundred RFQs is 5 seconds. For an organization that processes a few hundred, or even a few thousand transactions per day, the extra seconds or minutes required might not have a significant impact.

One last issue regarding these tests is that the measurements reflect translation time based on a format acceptable to the translator. In an operational environment a user might need to develop reformating software (see sec. 3.5.1) which converts the application data to a format acceptable to the translator. This reformating software can degrade performance. The amount of degradation depends on the complexity of the reformating software, which, in turn, depends on the flexibility of the EDI mapping utility. A flexible mapper might require only a few changes to the application data format, whereas, a rigid mapper might require the data format to be completely restructured. The point is that the use of reformating software, which can significantly impact performance, is not reflected in the NIST performance tests.

4.2.3 Table-Driven Verses Compiled Translators

The four translators used in this experiment are all table-driven translators. With a table-driven translator, EDI standards are stored in indexed files and are accessed every time the translator is run. For users requiring optimal performance, table-driven EDI software might not be the best architecture.

An alternative architecture is a compiled translator.[3] Rather than having the EDI standards reside in indexed files, they are stored in the translator process (i.e., data structures in the binary code). Thus, accessing the standards involves only a reference to internal memory rather than reading from a disk.

Improved performance does not come without a price. Although a compiled translator should outperform a table-driven translator, the compiled translator is harder to maintain. For example, updates to EDI standards and new data mappings (i.e., adding support for a new document type) requires recompiling portions of the translator. Updating table-driven software involves only updating the indexed files containing the EDI standards; no recompiling is required.

[3] There are very few compiled translators available commercially. No compiled translator could be obtained for the ECIF performance example.

5 Integration and Implementation Issues

Understanding the functional and performance capabilities of EDI products, while useful, provides only part of the knowledge required to implement an EDI system. To take full advantage of the benefits of EDI, an EDI system must be integrated into both the current operations and the future direction of an organization. A company's long-term vision must be taken into account when implementing its EDI system.

This chapter, divided into three sections, highlights issues addressing the immediate implementation of an EDI system as well as incorporating EDI into a company's long-term information management plan. Section 5.1 discusses integrating an EDI product into the business process. Section 5.2 describes communicating with trading partners, and section 5.3 overviews issues specific to using EDI for purchasing within the Federal Government.

5.1 Integrating EDI Into the Business Process

EDI is a business tool with many documented benefits; it minimizes overhead costs, improves response time, reduces the need for paper, eliminates transcription errors, and many others. Maximizing the extent of these benefits, however, involves more than just the purchase of a translator and a modem. An analysis of the business process must accompany any EDI implementation strategy. Adding EDI to an obsolete business system might make the system even less manageable. Changes to current business applications and processes may be required to gain the benefits of a fully integrated EDI system.

The goal of such an integrated system is to have data passed to all relevant applications without human intervention. For example, a small hardware store installs a bar code scanner at the cash register to total the bill for a customer. The bar coding software also updates the inventory for each item purchased. If the inventory falls below a certain threshold, the bar coding software triggers an ordering system which creates a purchase order for the deficit items, translates the order into an X12 850, and communicates the order to a distributor (either directly or through an EDI VAN). No human intervention is required.

Analyzing the above scenario highlights two types of integration: integration between business applications and integration between an application and an EDI translator. In the example, the bar coding system and ordering system need to communicate. Integration of the two systems is simplified if they reside on the same machine. If the systems reside on separate machines, and especially if the machines are different hardware platforms, integration becomes a more complex task. In addition, the bar coding and ordering systems may each maintain a database. Sharing information between dissimilar databases presents additional implementation issues. This example shows that before EDI can be fully integrated into the business process, the flow of information within all relevant business systems must be examined.

The second type of integration illustrated in the example involves coupling EDI with the ordering system. The seamless integration of EDI is not a trivial task. If the ordering system writes purchase order data to a file, and this file is used as input to the translator (i.e., a flat-file), the format of the purchase order data will most likely need to be modified to accommodate the translator. There are two reasons for this requirement.

Most translators maintain a trading partner directory. Information needed to transmit an interchange to a trading partner, such as the partner's EDI name (e.g., DUNs number), the standard and version to use (e.g., X12 version 003010), whether to request an acknowledgment, and other information, is stored in this directory. Each trading partner in the directory is assigned a unique identifier. When sending a transaction to a trading partner, most translators require this unique identifier to be included in the flat-file.

The second reason for modifying the format of application data is to specify the start and end of a loop (e.g., the line items in a purchase order). A translator must understand how many iterations of a loop to perform. Most commercial translators mandate a specific syntax for items in a loop, such as requiring all items comprising one iteration of a loop to be placed on one line in the flat-file. If a business application lacks the flexibility to accommodate these special types of requirements, a user might need to write software which converts the application data into a format acceptable to the translator.

The example of the hardware store illustrates the need to analyze the flow of data within a company's internal systems before implementing EDI. Another aspect of this analysis concerns the scope of the proposed EDI system. An EDI system can be implemented as a centralized or distributed service. With a centralized service, one EDI translator supports multiple applications. For example, an insurance company operates three business applications: health care claims, procurement, and finance. These applications may reside at one office or may be spread over multiple offices. In a centralized environment, one EDI translator supports transactions from all three applications. This approach is often less expensive and quicker to implement than its distributed counterpart.

In a distributed environment, each application uses its own translator. For example, if the insurance company depicted above chooses to implement a distributed EDI system, the company would maintain three translators: one for each of its health care claims, procurement, and finance applications. Although implementing multiple translators is more expensive than implementing one centralized translator, there are advantages to the distributed approach.

One advantage is ease of integration. Integrating a translator with a co-resident application is easier than sending application data from a different computer system. This is especially true if the application resides on a different hardware platform than that of the translator.

Co-residence between the application and translator also offers security advantages. If a purchase order is to be signed, the signature cannot be generated until after EDI translation (this is because the translation process alters the data, invalidating any signatures calculated on the application data). If a translator supports multiple applications, the EDI system must manage the keys for each application. This is highly undesirable from a security standpoint.

One final advantage to a distributed EDI system is that it follows an industry trend. Many commercial business applications now include an EDI component. For example, many federal purchasing packages contain an integrated EDI translator. If a company decides to replace a business application, the integration of EDI might already be provided. One warning regarding these EDI-enabled applications is that the EDI components might only implement a subset of the standard EDI transactions. For example, an EDI translator included with a federal purchasing package might only support procurement-specific EDI transactions. This warning is not intended to suggest that limited transaction support is undesirable, but simply to highlight that the EDI component of EDI-enabled applications might not support the full spectrum of EDI transactions.

5.2 Communication With Trading Partners

If a user purchases an EDI product that does not provide communications software, the translator must be integrated with some communications system. This will most likely involve using asynchronous or bisynchronous protocols and an EDI VAN.

The integration of a translator with modem software is not a complex task. The translator writes the EDI data to a certain file, and the modem software uploads that file to a VAN. For inbound transactions, the modem software calls the VAN and writes the received EDI data to a certain file. The translator uses that file as input. The most complicated task in this process is constructing the VAN script. The VAN script governs the timing and the content of the information exchanged between the user's computer system and the VANs computer system. Examples of this exchange include login (i.e., mailbox) identifiers, passwords, and the actual interchanges that are uploaded and downloaded. Some EDI translators that provide communications software include scripts for all major VANs.

Whether a user buys a translator with a communications package or integrates a translator with existing modem software, there are a few VAN issues which must be noted. The user must ensure that the VAN supports the protocol being used. Not all VANs, for example, support the X.25 protocol. The user must ensure that the VAN supports the standards and versions being used. Although this paper only references ASC X12 and UN/EDIFACT, there are other syntaxes (e.g., TDCC, VICS, and others) that might require support. Some VANs restrict the data segment and data element delimiters that can be used. This can cause conflict if the VAN does not support a delimiter needed by a trading partner. Finally, the user must ensure that the VAN supports the access method desired by the user. Although all VANs support a dial-up line, the user might require this line to be toll free, or might require a leased line.

As technology advances, some EDI users will replace traditional EDI bulk data transfer protocols (e.g., XMODEM) with more sophisticated communications services, such as electronic messaging. In this scenario EDI becomes one message type in a company's messaging system. Other message types include:

Interpersonal Messages: Often referred to as *email*, these are the memo-like messages exchanged between people.

Graphical Messages: These include pictures, spreadsheets, and other information types that are exchanged as binary data.

Voice Messages: Not to be confused with leaving a message on a telephone answering machine, these are messages that are played on a computer's audio device rather than being displayed to the screen.

Thus, a company might need to review its corporate messaging strategy, and regard EDI as one component in an overall messaging infrastructure.

5.3 EDI for Federal Procurement

The use of EDI for federal procurement is specified in a document called *Streamlining the Federal Procurement Process*[2]. The document defines an architecture comprising four major components:

Procurement Office: This component is responsible for all purchasing and financial functions.

Electronic Commerce (EC) Gateway: This component acts as an intermediary between the procurement office and the Network Entry Points (NEPs). Its primary functions are EDI translation, communications, and security.

NEP: This component acts as a intermediary between the Federal Government (specifically, the EC Gateways) and the EDI VANs.[4] Its primary function is to provide a central location to which federal purchasing transactions can be sent. A NEP forwards these transactions to the appropriate VAN, or in the case of a RFQ, to **all** VANs. The set of NEPs used for government procurement transactions is called the Federal Acquisition Computer Network (FACNET).[5]

VAN: This component provides a communications network for the Federal Government and its trading partners (i.e., federal suppliers).

The FACNET architecture is depicted in figure 3.

The current implementation of FACNET consists of two NEPs owned and operated by the Department of Defense (DoD). These NEPs initiate communications with all registered EC gateways and all certified VANs in a round-robin fashion. This means, for example, that each gateway and each VAN might get called by the NEPs once per hour.

[4] In this context, VANs are limited to those organizations that have completed the government VAN certification process.

[5] FACNET is sometimes described as the set of NEPs *and* EC Gateways used for federal procurement transactions.

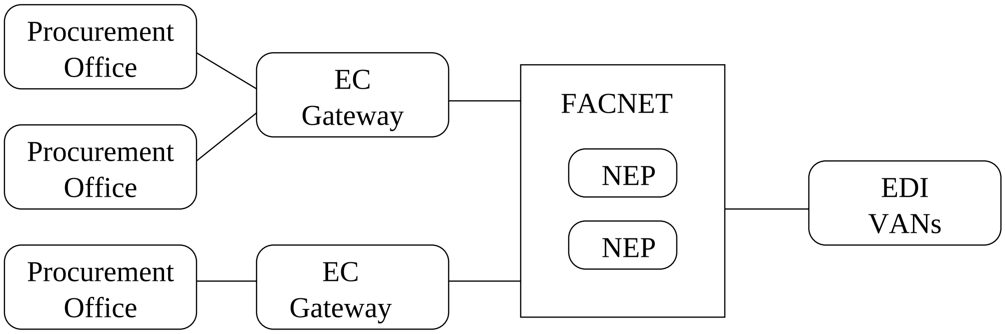

Figure 3: FACNET Architecture.

Using FACNET enables federal procurement offices (or gateways) to reach any or all certified VANs. Several implementation issues regarding this service need to be highlighted. These issues relate to communications and EDI naming conventions.

The first issue pertains to communicating with a NEP using modem software. As previously stated, a NEP initiates the communication with an EC gateway. This means the communications service implemented at the EC gateway must operate in a mode which waits to be called. Although many EDI translators include communications software, not all these communications packages have the capability to receive calls. Some only offer the capability to initiate a connection to a VAN or trading partner.

When communicating with a NEP, an EC gateway needs to inform the NEP which VAN or VANs are to receive the transaction. This information is provided via transaction filenames. For example, if an EC gateway has a transaction destined for a trading partner on the AT&T VAN, the name of the file containing the transaction would have the first three letters of *att*. Each VAN accessible via FACNET is assigned a specific set of letters for this purpose. In addition, there is a set of letters used to inform a NEP that a transaction (e.g., an RFQ) is to be communicated to all VANs.

When transmitting RFQs through FACNET a federal procurement office has the potential of receiving a multitude of bids. Some of these bids might originate from suppliers registered to conduct business with the Federal Government, but who have never conducted business with the requesting procurement office. Since most EDI translators only accept transactions from recognized trading partners, a user purchasing an EDI product for use in federal procurements must ensure that the translator is capable of accepting transactions from any registered supplier.

The final issue relating to federal procurement is the placement of EDI names in the X12 interchange. In a typical EDI transaction, the EDI names (e.g., the DUNs numbers) for the sender and recipient are placed in the ISA header of the interchange, and the interchange is sent to a VAN. Using FACNET, the interchange is not sent directly to a VAN, but to the NEPs. As a result the EDI name for the NEP needs to be included in the interchange. This is accomplished as follows:

1. The EDI name for the federal trading partner is placed in the *Interchange Receiver ID* field within the ISA header.

2. The EDI name for the NEP is placed in the *Interchange Sender ID* field within the ISA header.

3. The EDI name for the federal trading partner is repeated in the *Application Receiver's Code* within the GS (i.e., functional group) header.

4. The EDI name for the procurement office is placed in the *Application Sender's Code* within the GS header.

When a NEP transmits an interchange to a VAN, the interchange is addressed as if the NEP originated the transaction. When a federal supplier returns a transaction (e.g., a bid), the supplier's EDI software reverses the sender and recipient fields in the ISA and GS headers, and sends the transaction to its VAN. The VAN forwards the transaction to the NEP. The NEP, which maintains a directory of EDI names for federal procurement offices, parses the first GS header to determine the destination procurement office, and forwards the transaction to the appropriate procurement office (or EC gateway).

When conducting business using FACNET, federal users must ensure that their EDI translators can support having the sender and receiver EDI names in the ISA header differ from those in the GS header. Some translators might simply copy the ISA sender and receiver identifiers into the GS header. This approach is not acceptable when transmitting interchanges through FACNET.

6 Summary

The benefit of using EDI can be described as *reducing the cost of doing business*. This benefit is often divided into the various aspects of cost savings, such as improved response time, reduced need for paper, and minimized overhead expenses. To take full advantage of these benefits, an organization must formulate its EDI requirements prior to implementing an EDI system. EDI requirements fall into three categories: functionality, performance, and integration.

EDI functional requirements define the set of actions to be performed by the EDI system. Formulating these requirements may be difficult, especially for a user unfamiliar with the technology. To address such a concern, this paper provides a comprehensive listing of the functions that may be provided by *any* EDI translator. Based on this listing, a user can create a set of mandatory or desired functions, and confirm that any EDI translator being evaluated for purchase supports them.

EDI performance requirements define the effectiveness of the operation of the EDI system. For example, a user might require a translator to process a given number of transactions over a given time period. Not only will this requirement vary with different users, some users will have no performance requirements (i.e., all commercial translators will support their performance needs). This paper provides some examples to assist a user with determining if performance should be a requirement, and defines a procedure for evaluating the performance of EDI translators.

The final EDI requirement pertains to integration. To take full advantage of the benefits of EDI, an organization must analyze the flow of data within current business systems and the proposed EDI system. The analysis should yield a design for an integrated system where data is passed to all relevant applications without the need for human intervention.

In conclusion, by examining EDI functional and performance requirements, a user can purchase the EDI translator that best meets the user's needs. By addressing EDI integration issues, a user can maximize the benefits associated with EDI.

A ECIF Overview

The Electronic Commerce Integration Facility (ECIF) is a program within the Computer Systems Laboratory (CSL) of the National Institute of Standards and Technology (NIST). The vision of the ECIF is to help realize a national electronic marketplace; one that is secure, affordable, open to all, and easy to use. Such a marketplace will promote the rapid economic growth of American industry.

There are many technical barriers inhibiting a national electronic marketplace. These barriers include:

- lack of secure electronic transactions,
- lack of enabling services and infrastructures (e.g., directory services),
- lack of interoperable communications, data management, and security services, and
- lack of information sufficient for users to invest in existing and advanced information technologies.

To help overcome some of these barriers, NIST has initiated a project called the Federal Procurement Testbed. The Federal Procurement Testbed is a collaborative effort with industry to create a laboratory environment at NIST where vendors can deploy procurement applications and electronic commerce infrastructure services (e.g., EDI translators and communications packages). The goals of the procurement testbed are:

- to promote the adoption of standards (e.g., EDI) to allow applications to interoperate,
- to demonstrate the benefits of a *plug-and-play* (i.e., integrated) electronic commerce infrastructure, and
- to assist information technology users identify and understand technology trends.

More than 20 vendors have participated in the procurement testbed. These vendors are listed in appendix B. NIST personnel have configured and integrated the vendor products, and using EDI Value Added Networks (VANs), constructed testing scenarios showing product interoperation. The test scenarios, as well as individual vendor products, have been demonstrated to employees of various federal agencies.

B Participating Vendors

The National Institute of Standards and Technology (NIST) wishes to acknowledge and thank the vendors who provided software and/or services to assist this project.[6] These vendors are:

- AT&T
- CompUSA
- Compusearch Software Systems
- Delrina Corp.
- EDI Solutions, Inc.
- Fischer International Systems Corp.
- Fisher Scientific Company
- GE Information Services
- Government Technology Services, Inc.
- Hewlett-Packard Company
- Interactive Catalogs, Inc.
- ISOCOR
- Loren Data Corp.
- Novell, Inc.
- Oracle Corp.
- Paperfree Systems, Inc.
- Procurement Automation Institute (PAI)
- St. Paul Software, Inc.
- Sterling Software
- Supply Tech, Inc.
- Trade Service Corp.
- TransSettlements
- Trinary Systems Inc.
- Unipalm Limited

[6] This acknowledgment does not imply recommendation or endorsement by NIST, nor does it imply that the products identified are necessarily the best available for the purposes of the reader.

C RFQ Test Data

The performance example provided in section 4.2 uses the four EDI translators installed in the Federal Procurement Testbed. Since no application is integrated with each of the four translators, application data was taken from examples in the Federal EDI Implementation Conventions. This appendix provides, in table format, the application data used for the performance example.

Before presenting the data, three issues must be discussed. The first issue pertains to the format of the application files. The second issue pertains to the information stored in the application files, and the third issues pertains to looping.

There is no flat-file format that is acceptable to all the translators used in the performance example. As a result, a format was selected that, with only minor variations, could be tailored to each of the translators. This format is to place each data item on a separate line. For example, if an RFQ contains the name, telephone number, and FAX number for a buyer, the buyer's name begins one line (e.g., line 10 in the application file), the buyer's telephone number is placed on line 11, and the buyer's FAX number is placed on line 12.

The second issue pertains to what information needs to be contained in the application files. EDI segments often require two data elements to identify and specify one piece of information. For example, the number 301 555 1212 can be a telephone number or a FAX number. X12 uses two data elements to convey the required information: the first element is a code that states the number is a telephone number, and the second element is the telephone number itself.

There are two methods by which a user can convey that a specific number in an application file is a telephone number. The first is to place the X12 code into the application file. In this case two data elements are needed in the application file (e.g., line 20 is the code identifying a telephone number and line 21 is the telephone number). The second is to configure the code into the EDI translator. For example, the translator knows that line 20 of the application file contains a number. The translator could be configured to always treat the number in line 20 as a telephone number. In this case only one data element would appear in the application file (i.e., the telephone number), and the code identifying the number as a telephone number would be *hard-coded* in the translator.

The example of a telephone number is trivial. Telephone numbers and FAX numbers typically have unique positions in a form (i.e., in a form viewed by a user a box will be labeled as *telephone number*). Other examples are not so well defined. For example, a part number in an RFQ line item may be a vendor part number, a manufacturer part number, or a national stock number, each of which is identified by a different code. Since some codes need to be included in the application files, it was decided that all codes would be included in the application files. This means that no application data is hard-coded into any of the translators.

The final issue pertains to looping. Some portions of a form may have a variable number of entries (e.g., the line items in an RFQ). There are two methods by which entries, such as line items, can be handled. The first is to configure the translator to treat the line item section as a loop (i.e., there may be zero or more entries in the application data file). An alternative approach is that if the application uses a form that can only accommodate a limited number of line items (e.g., 10), to configure the translator to treat the line item section as 10 individual sections. In the case where there are less than 10 line items for a specific RFQ, the application file is padded with blank lines which serve as placeholders for the empty line items. For the performance test example, the translators were configured to use looping wherever the X12 standard specified a loop condition.

Having defined the format of the application file and the data to be included in it, the information is now presented in the form of a table. The table contains four columns. The first column identifies the X12 data element to which the application data element is mapped. The second column contains the value for that element as specified in the simple RFQ example and the third column contains the value for that element as specified in the complex multiline RFQ example. Blank table entries represent blank lines in the application file. The final column provides a terse description of the data.

Element	Simple RFQ	Complex RFQ	Description
BQT01	00	00	an original solicitation (00)
BQT02	N0001994B2055	N0001994R2055	the solicitation number
BQT03	940401	940401	the solicitation date
BQT07	03	04	the solicitation is an IFB (03) or RFP (04)
REF01	73	TN	code for a statement of work (73) or transaction set (TN)
REF02	12345	N000192252055	the statement of work or transaction set number
REF01	IJ	IJ	code identifying a SIC code (IJ) applicable to RFQ
REF02	56789	12345	the SIC code
REF01	DG		code identifying a drawing (DG) applicable to RFQ
REF02	22222		the number of the drawing
PER01	HM		a hazardous material (HM) contact
PER02	SAM SMITH		contact name is Sam Smith
PER03	TE		code identifying a telephone number (TE)
PER04	1234567890		the telephone number of Sam Smith
FOB01	PP		transportation paid by selling party (PP)
FOB02	DE		paid to the FOB point, destination (DE)
FOB06	DE		acceptance is at destination (DE)
CSH01	IS	K	a substitute item is allowed (IS)
			small purchase set aside for small business (K)
CSH06		AX	code for X12 as association assigning code values (AX)
CSH07		IO	items will be inspected at origin (IO)
CSHO8	X		substitute must be name brand or equal (X)
DTM01	997	997	code identifying responding quote date (997)
DTM02	940430	940430	the responding quote date
DTM01		996	code identifying delivery date (996)
DTM02		940617	the delivery date
PID01		F	code identifying a free-form description (F)
PID05		LANDING GEAR	free-form summary of the solicitation
MEA01		CS	the MEA segment states:
MEA02		MX	the maximum (MX) number (CS) of employees (IE)
MEA03		499	for the business responding to the RFQ
MEA04		IE	can be 499
PWK01		MR	code for material inspection and receiving report (MR)
PWK02		WS	reports to be sent with the shipment (WS)
PWK03		6	6 copies are to be sent
PWK08		2	code stating the reports are to be filed (2)
PWK01		C1	code for cost and price data (C1)
PWK02		EL	data must be provided electronically (EL)
PWK08		5	bidder must generate data and send with quote (5)
RRA01		14	code citing a clause pertaining to the response (14)
RRA02		52.203-2	the clause number
N901		DF	code citing a DFAR clause (DF)
N902		252.223-7001	the DFAR clause
N901		DF	code citing a second DFAR clause (DF)
N902		252.219-7000	the second DFAR clause

Element	Simple RFQ	Complex RFQ	Description
N101	BY	BY	code identifying the buying party (BY)
N102			
N103	10	10	buyer's address provided by DODAAC (10)
N104	N00288	N00288	the buyer's DODAAC
N301			
N302			
N401			
N402			
N403			
PER01	BD	BD	code for the bill-to party for diversion charges (BD)
PER02	BOND, JAMES	BOND, JAMES	party's name is James Bond
PER03	TE	TE	code identifying a telephone number (TE)
PER04	3015551111	3015551111	the telephone number of James Bond
PER05		FX	code identifying a FAX number (FX)
PER06		3015552222	the FAX number of James Bond
N101	PK	PK	company receiving the RFQ (PK)
N102			
N103	1	1	address is indicated by a DUNS number (1)
N104	130006789	1300006789	the DUNS number
N301			
N302			
N401			
N402			
N403			
PER01			
PER02			
PER03			
PER04			
PER05			
PER06			
N101		KY	code for a technical office (KY)
N102		AVIATION TECH	technical office that is supporting the RFQ
N103			
N104			
N301		3461 LIND WAY	street address of technical office
N302		SUITE 501	street address of technical office
N401		SYRACUSE	city of technical office
N402		NY	state of technical office
N403		08691	zip code of technical office
PER01			
PER02			
PER03			
PER04			
PER05			
PER06			

Element	Simple RFQ	Complex RFQ	Description
P0101	0001	0001	line item 0001
P0102	10	4	line item amount
P0103	ST	ST	line item unit is for sets (ST)
P0106	FS	MG	code for NSN (FS) or manufacturer's part number (MG)
P0107	82145B51B3587	B918273645	the national stock number or part number
P0108	CN		code for commodity name (CN)
P0109	SPARK PLUGS		the name of the commodity
MEA02			
MEA04			
MEA05			
MEA06			
MEA02			
MEA03			
MEA04			
MEA02			
MEA03			
MEA04			
PID01		F	code identifying a free-form description (F)
PID05		AIRCRAFT BRAKES	line item 0001 is for aircraft brakes
PWK01	MR	PD	receiving report (MR), proof of delivery (PD) required
PWK02	WS	BM	sent with shipment (WS), sent by mail (BM)
PWK03	6	1	number of copies
PWK08	2	2	code stating the reports are to be filed (2)
PKG01		F	free-form description of packaging (F)
PKG05		BUBBLE WRAP	line item 0001 is to be bubble wrapped
P0404			
P0410			
P0411			
P0412			
P0413			
REF01		IL	code for government internal requisition number (IL)
REF02		V0006540210003	line item 0001 is based on this number
IT801			
IT807			
FOB01	PP		transportation paid by selling party (PP)
FOB02	DE		paid to the FOB point, destination (DE)
FOB06	DE		acceptance is at destination (DE)
SDQ01			
SDQ02			
SDQ03			
SDQ04			
SDQ05			
SDQ06			
TD504		T	shipped by shipper's option (T)
MAN01			
MAN02			
LDT01	AF		line item delivered within (AF)
LDT02	90		90
LDT03	DA		calendar days (DA) of resultant purchase order
N101	ST	ST	code identifying where material is to be shipped (ST)
N103	16	16	code identifying zip code (16)
N104	78256	78256	the zip code

Element	Simple RFQ	Complex RFQ	Description
P0101		0002	line item 0002
P0102		20	line item amount
P0103		EA	line item unit is for “each” (EA)
P0106		FS	code for national stock number (FS)
P0107		3020005080982	the national stock number
P0108		ZB	code for cage code (ZB)
P0109		55555	the cage code of the manufacturerer
MEA02			
MEA04			
MEA05			
MEA06			
MEA02			
MEA03			
MEA04			
MEA02			
MEA03			
MEA04			
PID01		F	code identifying a free-form description (F)
PID05		GEAR BEVEL	line item 0002 is for gear bevels
PWK01			
PWK02			
PWK03			
PWK08			
PKG01			
PKG05			
P0404			
P0410			
P0411			
P0412			
P0413			
REF01		IL	code for government internal requisition number (IL)
REF02		V0006540210001	line item 0002 is based on this number
IT801			
IT807			
FOB01			
FOB02			
FOB06			
SDQ01		EA	the SDQ segment states:
SDQ02		10	line item 0002 is to be delivered to two
SDQ03		N00256	different locations identified by DODAACS (10).
SDQ04		10	10 EA to DODAAC N00256,
SDQ05		N00284	and
SDQ06		10	10 EA to DODAAC N00284
TD504		AF	deliver via air express (AF)
MAN01		L	the line item (L) packaging is to bear the marks
MAN02		FOR VF65	“for VF65”
LDT01			
LDT02			
LDT03			
N101			
N103			
N104			

Element	Simple RFQ	Complex RFQ	Description
P0101		0003	line item 0003
P0102		1000	line item amount
P0103		PR	line item unit is for pairs (PR)
P0106		VP	code for vendor part number (VP)
P0107		NB3421	the vendor part number
P0108			
P0109			
MEA02		PO	delivery quantity is permitted to vary by percent (PO)
MEA04		P1	unit of variance is by percent (P1)
MEA05		5	quantity can be 5 percent less
MEA06		5	quantity can be 5 percent more
MEA02		LN	line item 0003 must have a specific length (LN)
MEA03		3	the length is 3
MEA04		IN	the unit of length is inches (IN)
MEA02		DI	line item 0003 must have a specific diameter (DI)
MEA03		.25	the diameter is .25
MEA04		IN	the unit of diameter is inches (IN)
PID01		F	code identifying a free-form description (F)
PID05		NUTS AND BOLTS	line item 0003 is for nuts and bolts
PWK01			
PWK02			
PWK03			
PWK08			
PKG01			
PKG05			
P0404		UNT71	line item 0003 to be packaged in unit container (UNT71)
P0410		10	container length is 10
P0411		6	container width is 6
P0412		6	container height is 6
P0413		IN	container measurement unit is inches (IN)
REF01		IL	code for government internal requisition number (IL)
REF02		V0006540210002	line item 0003 is based on this number
IT801		IS	a substitute product is allowed (IS)
IT807		X	substitute must be name brand or equal (X)
FOB01			
FOB02			
FOB06			
SDQ01			
SDQ02			
SDQ03			
SDQ04			
SDQ05			
SDQ06			
TD504		T	shipped by shipper's option (T)
MAN01			
MAN02			
LDT01			
LDT02			
LDT03			
N101		ST	code identifying ship-to address (ST)
N103		16	code identifying a zip code (16)
N104		78256	the zip code
CTT01	1	3	number of line items

References

[1] Electronic Data Interchange (EDI). FIPS 161-1.

[2] Electronic Commerce Acquisition Program Management Office *Streamlining the Federal Procurement Process*; 1994, April.

Bibliography

American Business Computer, *EDI-ExCel User's Manual.*, Version 2.10, August 1994.

Cafiero, William, G., *A Plethora of Providers*, EDI World, 1994; February.

Derby, Brian, *Translation Software Steps Up EDI Integration*, EDI World, 1994; July.

EDI Solutions, Inc., *EDIeasy Entry Operator Manual.*, Revision 5.2, 1993.

EDI Solutions, Inc., *EDIeasy Technical Operator Manual.*, Revision 5.2, 1993

Electronic Data Interchange, ASC X12, Draft Version 3 Release 4 Subrelease 1; 1994, February.

Frohman, Harold, L., *A Guide to EDI Translation Software*, Logistics Management Institute; 1994, May.

Gant, David, R., *The Evolution of Communication Protocols in EDI*, EDI World; 1992, January.

An Introduction to Electronic Data Interchange, DISA Publications Catalog; 1993, January.

Ward, Kay, *EDI and Translation Software Products*, EDI Forum, The EDI Group, Ltd.; 1991, no. 2.

PaperFree Systems, Inc., *WWIX-MAP Reference Manual.* Copyright 1992, 93.

PaperFree Systems, Inc., *WWIX-MAP Tutorial.* Copyright 1992, 93.

Pittman, Lee, *How to Evaluate EDI Software*, EDI World; 1993, July.

Saltman, Roy, *Issues in Adoption of UN/EDIFACT International EDI Standards* (draft); 1994, September.

Supply Tech, Inc., *STX for the Microcomputer Basic User Manual.* Fourth Edition, September 1993.

Supply Tech, Inc., *STX for the Microcomputer Overlay Development Tutorials Manual.* July 1992.

Supply Tech, Inc., *X-MAP Reference Manual.* First Edition, January 1994.