Jointly with Book Industry Study Group (US)
and Book Industry Communication (UK)

FTP FILENAMING STANDARD – Issue 4

Changes from Issue 3 are shown in red.

BACKGROUND AND PRINCIPLES

1. The purpose of these filenaming conventions is to enable FTP file transfers of a variety of types of files, particularly EDI transaction files and book trade product metadata files, and including zipped files, to be made safely and securely. (They apply only to point-to-point transfers between trading partners. When FTP is used to send or receive files to or from a network operator, filenaming conventions are normally specified by the operator.)

2. The conventions defined here are based on a standard previously developed by BISAC (formerly BASIC), the US book trade standards group (part of BISG), and extended by collaboration between the BISAC Supply Chain EDI Committee and the BIC Technical Implementation Clinic. At this stage, therefore, the conventions cover national EDI formats used in the US and UK book trades together with established international formats supported by EDItEUR. It is open for other national groups in EDItEUR to propose extensions that may be needed for local use.

3. These conventions enable the file name to identify the format standard and the transaction type, but these features are not mandatory if trading partners agree that they will not be used. The status of the transfer is also encoded in the file name, and this encoding is mandatory.

4. It is a requirement of these conventions that both parties to a transfer should have access to the FTP server with the ability to change a filename.

PROCESS

5. The process of FTP transfer in accordance with this standard involves the following stages. At each stage only the party which currently "owns" the file may perform any operation on it.

(a) The sender puts the file on to the server. While this operation is in progress, the filename carries a status code indicating that the file is not yet ready for extraction and processing. At this stage, the file is "owned" by the sender.

(b) The sender completes the “put”, and changes the status code to indicate that the file is ready to be extracted. This signals that the sender has transferred “ownership” to the receiver.

(c) The receiver starts to extract the file, and, if the process of extraction so requires, may change the status code to indicate that this operation is in process. On successful completion, the receiver changes the status code to indicate that the transfer has been successfully accomplished, but the file has not yet necessarily been processed. "Ownership" remains with the receiver.

(d) When the file has been processed, the receiver changes the status code to indicate that the file can now be deleted from the FTP server. "Ownership" passes to whichever of the parties is responsible for managing the server.

FILENAME SPECIFICATION

6. The filename consists of a unique name and an extension. All letters of the alphabet must be in upper-case to avoid problems with systems that are case-sensitive.

7. There are two alternatives for the name: 8-character, and extended. The 8-character format is limited to eight alpha-numeric characters for the name. The extended format allows the name to have any reasonable number of alphanumeric characters, provided that the final part of the whole string is a three-letter extension as specified below. The filename may have multiple subdivisions with any suitable separator character (including, in systems which use this convention, a full point or period, as in, for example, ABC123.BOI.OCT23; but this is NOT an EDItEUR recommendation). The name must be unique for transfers between the two trading partners concerned.

8. The extension is always three letters preceded by a full point or period. The first letter is the *Status Byte*, indicating the status of the file transfer. The second letter is the *Format Indicator*, specifying the format standard used in the file. The third letter is the *Transaction Type* code, indicating the type of document or transaction message that is carried in the file.

STATUS BYTE CODE VALUES

9. The following values are currently assigned as *Status Byte* codes. The party responsible for setting the code value is shown in the right-hand column.

X	Creation or transmission in process	Sender
C	Creation/transmission complete – ready for pick-up, unzipped	Sender
Z	Creation/transmission complete – ready for pick-up, zipped	Sender
R	Extraction in process	Receiver
E	Extraction complete – unzipped	Receiver
Y	Extraction complete – zipped	Receiver
A	Processing complete – ready to remove from server and archive	Receiver

The use of code value R is optional, at the receiver's discretion. Its purpose is to allow files that are in process to be distinguished from newly arrived files, if and only if the receiver's extraction process so requires.

FORMAT INDICATOR CODE VALUES

10. The following values are currently assigned as *Format Indicator* codes. Additional codes will be defined when new standards are adopted, eg for XML-EDI transactions.

X	BASIC (BISAC or SISAC) X12 (US)
D	EDITX XML (international)
E	EDItEUR EDIFACT (international)
F	BISAC Fixed Format (US)
M	MARC (international)
O	ONIX XML (international)
T	BIC TRADACOMS (UK)
P	Proprietary: any format that is not a recognized standard to which a Format Indicator code has been assigned
Z	Undefined: use when trading partners have agreed that there is no requirement that the format should be stated in the filename

TRANSACTION TYPE CODE VALUES

11. The following values are currently assigned as *Transaction Type* codes. Additional codes will be defined when new transactions are added.

A	Order Acknowledgement or Response	X12 855, EDIFACT ORDRSP, TRADACOMS ACKMNT, EDItX OrderResponse and OrderStatusReport (see also D, below)
B	Bill & Ship Notice	X12 857
C	Chain Store Addresses	X12 885
D	Order Fulfilment	EDIFACT ORDRSP, when used in library supply for "order fulfilment" messages sending copy detail added by the supplier
F	Functional Acknowlegement	X12 997, EDIFACT CONTRL
I	Invoice/Credit Note/Debit Note	X12 810, EDIFACT INVOIC, TRADACOMS INVFIL and CREDIT
J	Journal Claims	EDIFACT OSTENQ
K	Journal Claims Response	EDIFACT ORDRSP
L	Statement/Remittance Advice	TRADACOMS SRMINF
M	Stock Enquiry	EDItX StockEnquiry
N	Stock Report	EDItX StockReport
P	Purchase Order (including Order Cancelation, Order Chaser)	X12 850, EDIFACT ORDERS and OSTENQ (as order chaser), TRADACOMS ORDERS and BTOERS, EDItX Order, CDFOrder, LibraryOrder, OrderStatusEnquiry, OrderCancelation
Q	Quote	EDIFACT QUOTES when used in library supply for supplier-initiated quotation message
R	Sales Reporting	X12 852, EDItX DigitalSalesReport
S	Ship Notice	X12 856, EDIFACT DESADV, TRADACOMS DELIVR, EDItX AdvanceShipNotice

/continued

Transaction Type codes (continued)

T	Title Status/Product Catalog	X12 832, EDIFACT PRICAT, TRADACOMS PVUINF, ONIX Product Information
V	Request for Returns Authorisation	EDIFACT RETANN
W	Returns Authorisation	EDIFACT RETINS (from supplier)
X	Returns Confirmation	EDIFACT RETINS (from customer)
Z	Undefined	Use when trading partners have agreed that there is no requirement that the transaction type should be stated in the filename.

ZIPPED FILES

12. It is an absolute requirement of these conventions that a zipped file should carry only one data file, to avoid confusion in filenaming and in any sequence numbering that may be carried in the filename. The zipped file and the data file should have the same name (eg zipped file DGA01167.ZXP, data file DGA01167.CXP)

13. It is assumed that the file is unzipped to a location determined by the receiver, and not on the FTP server.

14. The software at the receiving end needs to unzip the file before it can take any other action. Consequently it is essential that the *Status Byte* distinguishes between zipped and normal files. The distinction is also maintained after apparently successful extraction, in case it proves necessary to go back and re-process the file.

EXAMPLES OF COMPLETE TRANSFERS

		File "Owner"
EDIFACT invoice file, unzipped – 8-character name		
BGH00236.XEI	File "put" in progress	Sender
BGH00236.CEI	Ready for extraction	Receiver
BGH00236.REI	Extraction in progress	Receiver
BGH00236.EEI	Transfer completed	Receiver
BGH00236.AEI	Processed and may be deleted	Server manager
ONIX product information file, zipped – extended name		
JBCDF7631204.XOT	File "put" in progress	Sender
JBCDF7631204.ZOT	Zip file ready for extraction	Receiver
JBCDF7631204.YOT	Zip file transfer completed	Receiver
JBCDF7631204.AOT	Processed and may be deleted	Server manager

In the first example the receiver chooses to change the status code to R while extraction is in progress. In the second example, the receiver chooses an extraction process that does not require this additional step.

APPENDIX: alternative methods of processing incoming files

During debates in creating this FTP filenaming standard, it became apparent that there were at least two major methods by which receivers commonly extract and process incoming files. While there might be variations on these themes, the two methods had clearly different requirements in terms of the use and interpretation of the *Status Byte*. Instead of mandating or recommending one or other of the two, the standard has been left open to both, and should work equally effectively with either.

The two methods are outlined below. By providing this added detail, attention is drawn to issues users will want to consider in determining their own particular implementation.

What the sender does is identical in both cases. The sender places files in a designated directory on the FTP server with names of the form *.X*, and upon completion of each transfer renames the transferred file from *.X* to *.C*

This process is completely independent of when and how the receiver chooses to access the server and extract files, and may be ongoing while the receiver is accessing the server.

Method One: using status code R to manage the extraction process

1. Receiver accesses the designated directory on the server, selects all files of type *.C*, and renames them generically to *.R*
2. Receiver processes the selected files as a batch.
3. Receiver renames all *.R* generically to *.E*

Any new files transferred by the sender while the receiver is processing the selected files will have names of the form *.C*, and there is no risk of confusion between a new file and one which has been extracted for processing.

Method Two: managing the extraction process without status code R

1. Receiver accesses the designated directory on the server, and captures a list of files of type *.C* at the moment of access.
2. Receiver works though the list of files one by one, and for each entry:
   2.1. GETs the file
   2.2. Processes the file
   2.3. Renames the file individually with a fully qualified name and with status code E

This method removes the need to use status code R, relying instead on a list of file names being kept in memory, and on renaming each file individually.

The critical thing to keep in mind is that these two methods must NOT be combined. If status code R is not used, files on the server must not be generically renamed to status code E, or new files which have arrived during processing will be renamed to E without being extracted.

[Note that for zipped files, status codes C and E are replaced by Z and Y respectively]