FF/EDM
Intro
Industry Goals/ Purpose
GISB defined two ways in which flat files could be used to send transactions and transaction responses: interactive and batch. This section covers implementation considerations for the use of interactive flat files.

In general, interactive flat file communication has similarity with EBB/EDM. For example, both involve human interaction and both use a Web browser to accomplish their purpose. Interactive flat files differ from EBB/EDM in how the transaction data is prepared. EBB/EDM allows for direct Web page entry of the data elements of the transaction while flat files are prepared as part of a separate process "off-line".

A variety of tools could be used to prepare flat files. However, what GISB had in mind was to facilitate the preparation by creating standards that are consistent with how spreadsheets can save files. Further, the standards were devised to avoid the need for programming (e.g., using spreadsheet macros) in order to create the file. The flexibility for the sender to order the data elements does imply programming to interpret the received file on the part of the recipient.

An interactive flat file process may choose different mechanisms to respond to the uploaded file. For example, the response may be an HTML screen which highlights any errors found or it may be a file response. Also, the response could be part of the same Web connection (HTTP round trip) or via an asynchronous mechanism (the user is either notified when the result is available or can go look for the result on a Web page).

This portion of the guide assumes an HTTP multipart form file upload. Other implementations (e.g., custom Java applet) are not described however some of the same considerations described below are applicable.

Related GISB Standards (Common Codes, IETF)
Definitions
d 4 - "GISB FF/EDM" is the term used to describe a standardized flat file electronic data interchange of information in files as mapped from the x.4.z GISB standards. GISB FF/EDM is communicated between trading partners over the Internet using the GISB Electronic Delivery Mechanism.
d 13 - "Interactive Flat File" is the term used within GISB FF/EDM to describe the transfer of flat files using an interactive browser.
Principles
p 6 - There should generally be a one-to-one relationship between data elements used for EDI and/or flat files and the data displayed on Customer Activities Web pages.
p 13 - For GISB FF/EDM, the content and usage of flat files should reasonably correspond to the GISB data sets used for GISB EDI/EDM.
p 14 - If GISB FF/EDM is implemented, flat files should be exchanged via the GISB EDI/EDM site or the Customer Activities Web site.
Standards
s 16 - Where they exist for the same business function, flat files and EDI should use the same nomenclature for data set names, data element names, code values and/or code value descriptions, abbreviations and message text. Corresponding Web pages should use data set names, data element names, code value descriptions, abbreviations and message text that correspond to those used in flat files and EDI, where they exist.
s 25 - GISB FF/EDM flat files should be formatted as ASCII comma separated value (CSV) files. This means:
Rows are separated by a carriage return/line feed (CRLF). Fields are separated by commas.
When a field contains a comma, the field should be enclosed by double-quotes.
Double-quotes should not be used within any data field.
When numeric data is negative, the minus sign should precede the number.
When numeric data contains decimal precision, the decimal point should be included within the field.
When numeric data contains one or more significant leading zeros, these zeros should be preserved in

the flat file.
Date fields should be formatted as YYYYMMDD.
Time fields should be specified in a 24 hour format, formatted as HH:MM or HH:MM:SS, as applicable.
Date/Time fields should be formatted as YYYYMMDD HH:MM or YYYYMMDD HH:MM:SS when date and time are expressed in one GISB data element. Note that there should be exactly one space between the day (DD) and the hour (HH).
The maximum amount of data to be placed in a field should be limited to 256 characters.
When a field contains no data, the empty field should result in two delimiters next to each other. Note that there should be no blank spaces between the delimiters.

s 27 - For a GISB FF/EDM flat file, the first row of the file should be comprised of the standard abbreviations for GISB data elements, including any additional data elements added per GISB Standard No. [s21], in the order in which the corresponding data is to appear in all subsequent rows. The data element order is at the option of the sender. If a data element abbreviation is not recognized, the entire flat file should be rejected.

s 28 - For GISB FF/EDM flat files, each transaction (e.g. nomination) should be contained in a single row.

s 29 - Where display information on a Customer Activities Web site is derivable from data provided in a previous upload or download, the information should not be included in the EDI/EDM standards [or FF/EDM standard, for later consideration] that directly correspond to the EBB/EDM Web page being displayed.

s 53 - For Interactive Flat File EDM, 40-bit Secure Sockets Layer (SSL) encryption should be used. Where possible, 128-bit SSL encryption is strongly recommended.

s 54 - Access to Interactive Flat File EDM should be protected by HTTP Basic Authentication.

Other Applicable Standards

HTTP Post with mult-part forms (RFC 1867)

Secure Sockets Layer (SSL) - HTTPS

Minimum Technical Characteristics of the Client Workstation

Flow Diagram

Specification

The Parts of a Page

General

While GISB did not explicitly determine the design of a Web page used for flat file uploads, it makes sense to achieve consistency with the design determined for EBB/EDM. This section follows the outline for EBB/EDM and makes suggestions as to how this consistency can be accomplished. This section applies to parties such as Transportation Service Providers (TSPs) which provide for the uploading of a flat file to its Web server.

Header Area

Left side

The top left side of the Web page should provide navigation to the Customer Activities home page and/ or directly to some of its major menu items. That is, it should look exactly like it does for this section for EBB/EDM.

Right side

The top right side of the Web page should provide for invocation of page functions as it does for EBB/EDM. Since uploading a flat file does not have need for most of the EBB/EDM functions, this portion of the page may be limited to such things as the “Submit” function.

Forms Area

The Forms Area will be uncomplicated for Interactive Flat File uploads. Its exact look will depend on how interactivity is implemented and whether optional response types are made available. At minimum, it needs to have a text box to specify the file to be uploaded. This text box will be accompanied by a “Browse” button to allow a graphical selection of the file versus having to type its full path and name. This button is provided automatically by the browser. It is also necessary to include a “Submit” button near (e.g., immediately below) the text box for the file name. This button is necessary as part of a multipart form. The “Submit” function mentioned above in the right side of the Functions Area could be made to programmatically (e.g., using Javascript) “click” this “Submit” button. If alternative response types (see Intro above) are provided, such choices could be made available with a drop-down list box. It may make sense to provide this ahead of (e.g., above) the text box which provides entry of the file name. Two other possible controls include a dropdown from which to choose the TSP being nominated and a text box to indicate the DUNS number of the nominator. These would simulate the "to" and "from" fields in

the batch EDM process. An example of what this may look like is provided in a subsequent section. As it is unlikely that this collection of user interface controls will require much screen real estate, it may make sense to allow a larger portion of the screen for response information if it is an HTML screen response.

Matrix Area

The matrix area could be used for an HTML response if that alternative is made available. If so, it is also desirable that it be as consistent as possible with the look and feel of the response resulting from EBB/EDM (assuming it is implemented on the site along with Interactive Flat file capability).

Page Functions

As was stated above, it is unlikely that there will be many functions besides the "Submit" function. The Submit function will have the effect of uploading the flat file for processing by the back end system. Depending upon the specific implementation, it may generate an acknowledgement of the receipt of the uploaded file, errors encountered in the prevalidation (if any) and/or the actual results of the backend processing (e.g., Quick Response info).

Page Format

It may be useful to implement two or more of the page sections as HTML Frames. To accomplish a file upload, the Forms Area must include a multi-part form which requires a special HTML values for the Form tag which are ENCTYPE="multipart/form-data", ACTION="scriptname" and METHOD="POST" where scriptname is the script or program which processes the upload file on the Web server. The form will also contain a tag specifying a file as a type of input such as the following: <input type="file" size="30" name="input-data">. It is this tag which causes the browser to create a text box and a button for browsing to a specific file. The GISB-specified browser release (i.e, version 4 or better) ensures that mulitpart forms are supported.

File Creation

As was mentioned in the Industry Goals section, it is envisioned that the creation of the required flat file format be possible without programming. Specifically, what the designers had in mind was the use of a spreadsheet to accomplish this. The user would first type a "heading" row which contains the names of the data elements being uploaded (see Standard s27). Then the user would type appropriate data values in subsequent rows of the spreadsheet (note Standard s28). When all data is entered, the user would choose a file save menu and choose a file type of "comma separated values". The user must carefully note where this file is saved so that it can be chosen in the browser Forms area as described above.

To facilitate the repeated use of this spreadsheet, it would make sense to save a spreadsheet in its native format which contains the heading information thus allowing reuse of this as a template for subsequent nominations. If this is done, the user must be careful not to choose this native format file (e.g., for Excel this would be the .xls file) as that to be uploaded as it will not be of the proper file type (it is a binary file and not the one with the necessary text layout). Other spreadsheet features may be employed to avoid having to repeatedly enter data (e.g., the contract identifier) which does not change from row to row.

While the vision is no programming, it does not preclude the use of macros or other "front ends" to make it easier for the user to create the proper file format. For example, a special program with a customized form for data entry could be written which facilitates easier data entry or integration with an existing system. This program would have the responsibility of taking the form data and arranging into a format compliant with the standard (see Standard s25).

Uploading Mechanism

If both EBB/EDM and Interactive FF/EDM are available, it may be useful to have submenus for each under the appropriate GISB standard menu. Once this menu is chosen, the user should be presented a Web page as described above under the Parts of a Page and Page Format sections.

Receipt Programming

Interpreting a multipart form upload

A multipart form is sent to the Web server using a layout described in the applicable Internet Request For Comment (RFC), currently RFC 1867. This RFC describes how a multipart form allows the uploading of a variety

of MIME types from a single form among them a File type. As part of the upload, an HTTP header is sent indicating the string of characters which acts as a delimiter for each part of the upload form. If the form is processed by a traditional Common Gateway Interface (CGI) program (e.g., using C/C++ or Perl or others), it will have to parse the data using the RFC as a specification of data format.

Using a commercial component to assist

For some Web servers it may be possible to obtain a commercially available component which reduces the task of receiving an uploaded file to simple object method and property syntax.

Assigning data element values (parsing the uploaded file)

Once the file has been successfully received by the Web server, it may be useful to pre-validate it as much as possible. For this to be done, the individual elements of the file need to be parsed and, presumably, saved to an array or data base table. Assigning the data elements to the proper storage area is facilitated by the first row which provides standardized abbreviations (see Standard s27) for each position in the delimited file's records (or rows). At this stage it may be possible to reject the uploaded file for various reasons thus avoiding sending "garbage data" to the backend system.

Pre-validations

At this stage it may be possible to reject the uploaded file for various reasons thus avoiding sending "garbage data" to the backend system. This could be the result of an unrecognized header row data element name. It could also be for such things as a file whose size is much bigger than can reasonably be processed by the backend. It may also be due to the discovery that the file is binary indicating a probable mistake by the sending party (e.g., upload of the spreadsheet's native format). In any case, the goal here is to avoid unnecessarily burdening the backend and providing the quickest possible response to the user.

Synchronous Vs Asynchronous

As was mentioned in the Industry Goals section, a variety of implementations are possible for Interactive Flat Files. One type of implementation could be characterized as "synchronous" where the user waits for the reply from the backend validations as part of the same HTTP round trip. In other words, after pressing the Submit button, the system returns receipt acknowledgement and the completely validated response to the browser which is waiting for that response.

A different implementation may only acknowledge receipt of the uploaded file and will make the results of the backend validation available some time later. The user may or may not be notified of the availability of the full validation response. If not, they may periodically check a particular Web link for a list of available responses. GISB was intentionally silent as regards how the EBB/EDM or Interactive FF/EDM accomplish showing validation results.

Yet other implementations may be possible.

Interface to backend system

GISB standards make no attempt to specify backend mechanisms so this is completely up to the individual providers. Typical implementations may include two-tier (traditional client/server applications), two-tier with data base stored procedures or three-tier. Again, other implementations are possible and this guide makes no attempt to be specific here.

Formatting the response

As mentioned above, the response can be an HTML screen or a flat file. This may be based on an option provided to the sender on the upload form. If it is a flat file response, it must conform to the GISB standards which include flexibility in the order of data elements within a record (or row). It may be more "user friendly" to have a well-defined (presumably published on the provider's Web site) sequence so to avoid making the user incur programming time and expense otherwise necessary to handle a variable sequence.

Examples

A typical spreadsheet

Possible HTML response

Security

Authentication

Standard s54 calls for use of Basic Authentication. This is a standard part of the HTTP 1.0 specification. Without use of encryption, this would be a clear text transmission of user id and password. To avoid this, merely protect the page from which the log on is invoked with Secure Sockets Layer encryption as described below. Note that where the user id and password information is maintained is different for different Web environments. You may want to consider providing the ability for users to change their password.

Encryption

Standard s53 calls for the use of 40-bit encryption using Secure Socket Layer (SSL) technology or equivalent. SSL is accomplished by obtaining a certificate from providers and using Web servers capable of using these certificates to accomplish SSL. The standard browsers specified in the Client Configuration standard are known to be able to handle SSL mechanisms. Any pages to be protected with SSL need to be invoked with the HTTPS protocol by using “https” versus “http” as part of the hyperlink (HREF) name. Note that this means using a Fully Qualified versus Relative link name. This, in turn, causes a new DNS lookup from the browser. When the hostname is provided by more than one machine, this may result in the request being sent to a different machine. This would only cause problems where necessary state information is being maintained in the memory of the Web server’s machine.