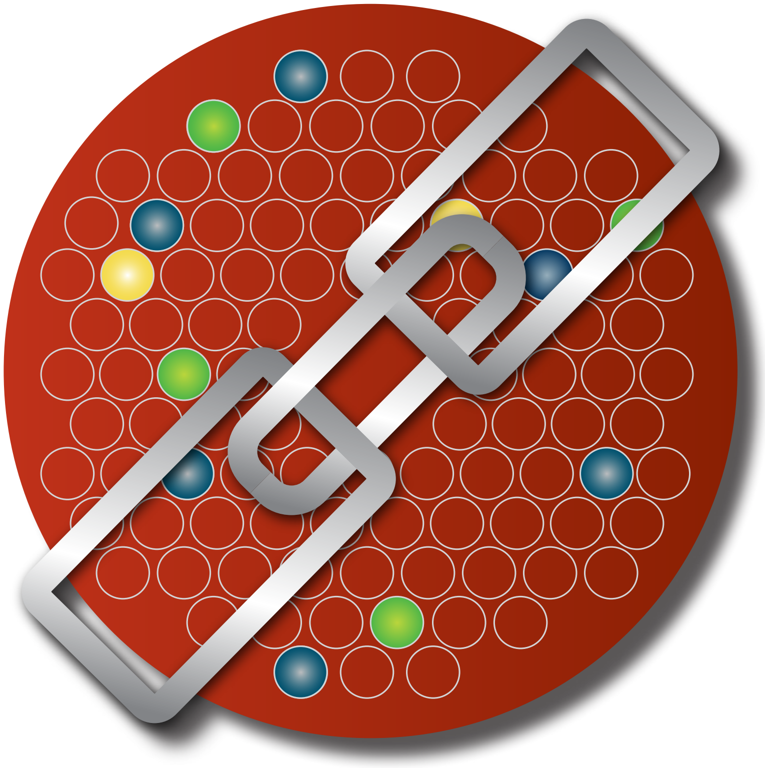
FilmArray® LIS Interface



Driver Development Guide

BioFire Diagnostics LLC
390 Wakara Way
Salt Lake City
84108
+1 801 736 6354
1 800 735 6544

BIOFIRE®
A BIOMÉRIEUX COMPANY

TABLE OF CONTENTS

CHAPTER 1: Introduction

System Overview

The FilmArray 2.0 system is a combination of FilmArray instruments, one analyzer (computer and software), and a reagent pouch (disposable). The system integrates sample preparation, nucleic acid amplification, detection, and data analysis into one platform, and is designed to process samples and deliver results in about one hour. It is intended to be used in combination with the FilmArray panels.

The design of the system ensures that tests are correctly performed and that test results are not misinterpreted. The FilmArray software plays a central role in achieving these goals. The FilmArray 2.0 system is controlled by Windows®-based software running on a connected computer. The FilmArray software is delivered as a single desktop application, referred to as the FilmArray Software, and panel-specific Pouch Modules. Each Pouch Module contains definitions required to run a pouch on a FilmArray instrument and analysis software to interpret the data from a run. The results are displayed as a PDF report in the FilmArray Software.

As an add-on to the system, the FilmArray Link Software and panel-specific Electronic Report Modules facilitate the unidirectional transfer of test results from a FilmArray 2.0 system to a laboratory information system (LIS). The FilmArray Link Software retrieves test result data from a configured FilmArray database, sends the data to the corresponding Electronic Report Module (where the data is formatted into a message, referred to as an Electronic Report), and then sends the Electronic Report to the bioMérieux Communication Interface (BCI) Link software. BCI Link transfers the Electronic Report to the LIS.

Purpose, Scope, and Audience

The purpose of this document is to provide an overview of the FilmArray 2.0 system as well as guidance regarding the development of the FilmArray LIS driver. The document specifies the format of the Electronic Report and the communication methods required to implement the FilmArray LIS interface. This document is intended to be used by LIS vendors during the development of the FilmArray LIS interface driver.

Reference Documents

The following documents are referenced in this document:

- ASTM 1394-97
- BCI Link User Manual
- FilmArray® 2.0 Operator's Manual
- ISO-8859-1
- RFC0959

Caution Statement

CAUTION: The computer and its operating system have been carefully configured for optimal performance of the FilmArray® 2.0 system. Altering the configuration may severely hamper the usability of the instrument. This document does not replace the Limitations of Use section of the FilmArray 2.0 Operator's manual.

CHAPTER 2: ASTM-XML Messages Format

Description

ASTM-XML messages are based on a subset of ASTM attributes defined in the specification ASTM 1394-97 and structured using XML formalism. To avoid any ambiguity in the interpretation of messages, an XML schema is used to check the syntax and ensure that the messages follow several semantic rules.

For further details on XML schema, refer to the Introduction to XML Schema below.

The purpose of the XML schema is to define the "`AIMessageType`" element for the definition of results.

Each XML element derived from the ASTM specification is preceded by a comment indicating the section in which the attribute definition can be found. For example, the definition of the patient identifier is presented as follows:

🕮 Chapter 8.1.4 of the ASTM specification describes the patient identifier

```
<!-- ASTM chapter 8.1.4 -->

<xsd:element name="patientIdentifier" type="StringL12"/>
```

ISO-8859-1 and UTF-8 type of character encoding should be used by default. The system only supports this type of character encoding.

IMPORTANT: Character encoding forms such as UTF-16 or UCS-2, which use two or more fixed bytes, are not supported by BCI Link for optimization reasons. Only variable width encoding forms which can code the first 255 characters on a single-byte are accepted by BCI Link (Shift-JIS, UTF-8, ISO-8859-1, etc.).

Note: The XML parser in charge of interpreting the bioMérieux result frame must manage:

- The CDATA section
- The 5 predefined XML entity references (&lt; &gt;&amp;&apos;&quot;) to replace the illegal XML characters
- The references to Unicode entities ("&#nnn;")

Introduction to XML Schema

XML schema is a language used to formally describe the content of an XML message. It is derived from the works of the World Wide Web Consortium (W3C); all the publications (specifications, tutorials, etc.) are available at http://www.w3.org/XML/Schema.

The following sections describe the notation used for the declaration of XML element types and cardinalities.

Declaration of Types

In XML schema, there is a basic difference between complex types, which allow elements in their content and may carry attributes, and simple types which cannot have element content and cannot carry attributes.

New complex types are defined using the *complexType* element and such definitions typically contain a set of element declarations, element references, and attribute declarations. The declarations are not themselves types, but rather an association between a name and the constraints which govern the appearance of that name in documents governed by the associated schema. Elements are declared using the *element* element and attributes are declared using the *attribute* element.

For example, *USAddress* is defined as a complex type, and within the definition of *USAddress* we see five element declarations and each element is simple type: (*string* or *decimal*):

```
<xsd:complexType name="USAddress" >
  <xsd:sequence>
   <xsd:element name="name"   type="xsd:string"/>
   <xsd:element name="street" type="xsd:string"/>
   <xsd:element name="city"   type="xsd:string"/>
   <xsd:element name="state"  type="xsd:string"/>
   <xsd:element name="zip"    type="xsd:decimal"/>
 </xsd:sequence>
</xsd:complexType>
```

Occurrence Constraints

An element is defined as optional by means of the value of the *minOccurs* attribute which is 0. In general, an element is required to appear when the value of *minOccurs* is 1 or more. The maximum number of times an element may appear is determined by the value of a *maxOccurs* attribute in its declaration.

This value may be a positive integer such as 41, or the term *unbounded* to indicate there is no maximum number of occurrences. The default value for both the *minOccurs* and *maxOccurs* attributes is 1. Thus, when an element is declared without a *maxOccurs* attribute, the element may not occur more than once. If both attributes are omitted, the element must appear exactly once.

FilmArray® XML Schema

The FilmArray® XML schema can be found in the XSD file attached to this manual. To access this XSD file:

1. Click 📎 on the left side of the screen.
2. Double-click on the XSD file to open it or right-click and to save it.

Message Structure

AI Message

The message is sent by the FilmArray® system to the LIS to report the result.

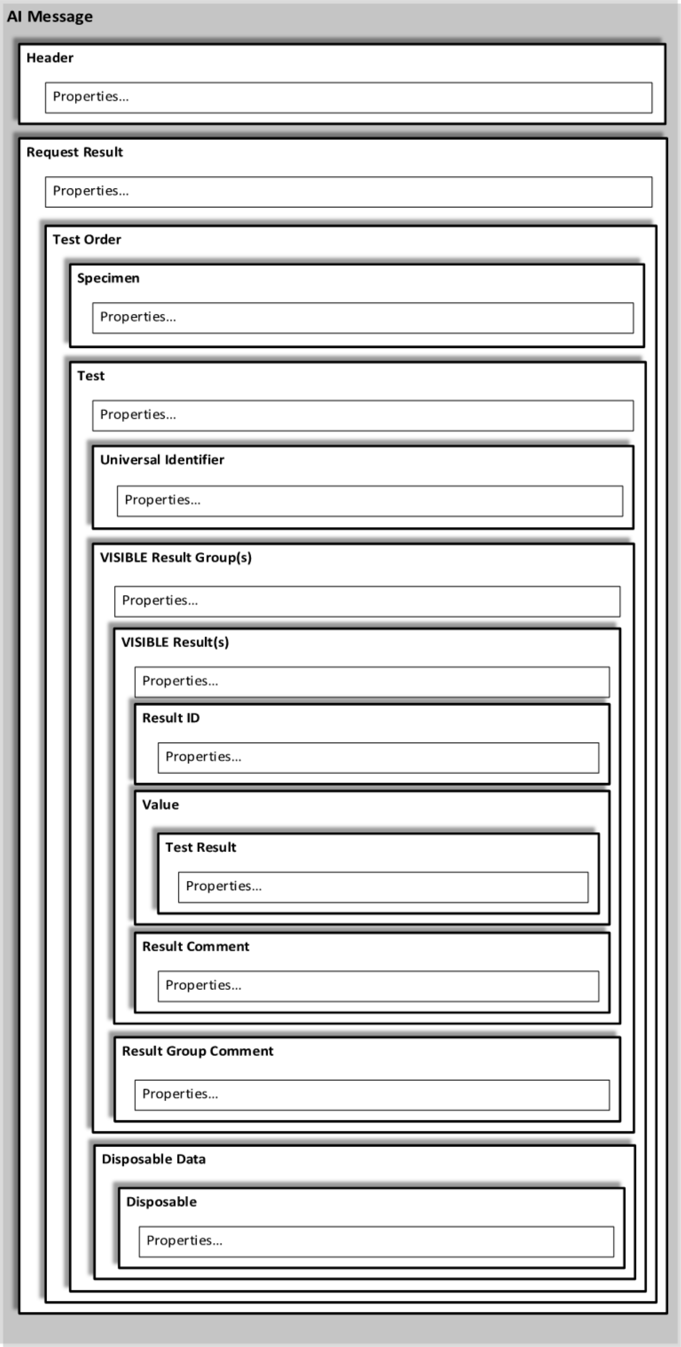


Figure 3-1: AI ASTM-XML Message Structure

There are XML files attached to this document and they include sample result messages.

1. Click 📎 on the left side of the screen.
2. Double-click on the file you want to open or right-click to save it.

Formatting Rules

Messages are formatted according to well defined formatting rules.

IMPORTANT: The decimal separator should always be "." (dot), whatever the locale is.

For more details on message format, see Data Types below.

Data Types

The table below lists all the possible fields with their associated characteristics.

Attribute	Value Type / Format / Constraints	Default Value	min:max Cardinality
/aiMessage/header			**1:1**
senderName	**StringL20**	FilmArray	0:1
processingIdentifier	**String**	P	0:1
version	**String**	1394-97	1:1
dateTime	**yyyyMMddHHmmss**		1:1
messageType	**StringL12**	FA_RESULTS	1:1
/aiMessage/requestResult			**1:N**
requestStatus	**StringL5**	F	0:1
/aiMessage/requestResult/testOrder			**0:N**
/aiMessage/requestResult/testOrder/specimen			**1:1**
specimenIdentifier	**StringL40**		1:1
/aiMessage/requestResult/testOrder/test			**0:N**
instrumentType	**StringL30**	FilmArray	0:1
instrumentSerialNumber	**StringL80**		0:1
/aiMessage/requestResult/testOrder/test/universalIdentifier			**1:1**
testIdentifier	**StringL20**		0:1
testName	**StringL60**		0:1
testVersion	**StringL6**		0:1
/aiMessage/requestResult/testOrder/test/resultGroup[n]			**0:N**
resultGroupCode	**StringL20**		0:1
resultGroupName	**StringL100**		0:1
resultGroupCodingSystem	**StringL40**	BMX	0:1

Attribute	Value Type / Format / Constraints	Default Value	min:max Cardinality
/aiMessage/requestResult/testOrder/test/resultGroup[n]/comment			**0:N**
text	**String**		0:1
/aiMessage/requestResult/testOrder/test/resultGroup[n]/result[n]			**1:N**
operatorName	**StringL30**		0:1
resultDateTime	**yyyyMMddHHmmss**		0:1
/aiMessage/requestResult/testOrder/test/resultGroup[n]/result[n]/resultID			**0:1**
resultTestCode	**StringL40**		1:1
resultTestName	**StringL255**		0:1
resultCodingSystem	**StringL40**	BMX	0:1
/aiMessage/requestResult/testOrder/test/resultGroup[n]/result[n]/value			**0:N**
/aiMessage/requestResult/testOrder/test/resultGroup[n]/result[n]/value/testResult			**0:1**
valueType	**String**	CE	1:1
observationValue	**String**		1:1
observationName	**StringL255**		0:1
/aiMessage/requestResult/testOrder/test/resultGroup[n]/result[n]/comment			**0:1**
text	**String**		0:1
/aiMessage/requestResult/testOrder/test/disposableData			**0:1**
/aiMessage/requestResult/testOrder/test/disposableData/disposable			**0:N**
disposableIdentifier	**StringL40**		0:1
reference	**StringL40**		0:1
disposableType	**StringL40**	Pouch	0:1
lotNumber	**StringL40**		0:1

CHAPTER 3: File Transfer Protocol

FilmArray® can transmit the test results using the FTP, as described in this section, or the Shared Folder protocol, as described in Shared Folder Protocol on page 20.

Analysis Result Naming Rules

Analysis result files sent by the AI is named as follows:

```
FILMARRAY_YYMMDD_HHMMSS_s.xml
```

Where:

- **YYMMDD** means Year, Month, Day.
- **HHMMSS** means Hour, Minute, Second.
- **s** is the sequence between 0 and 99. This is used if the AI sends more than one analysis result (in the same second).

Transport of ASTM-XML Messages

Messages are transported by TCP/IP using the standard FTP. From a strictly "communications" point of view, the BCI Link software is an FTP server and the LIS is an FTP client. In other words, the interface driver should integrate an FTP client who is capable of uploading files containing the results.

The FTP requires two connections for communication, a "control connection" and a "data connection". The control connection is used to exchange FTP commands and the data connection to transfer data.

The transmission mode is a "Stream" mode, i.e. the file is simply transferred as a series of bytes. There are no limitations for the data representation.

The “file type” structure is used. The EOF (End Of File) sequence is implicitly marked by the closing of the connection. All the transmitted bytes are therefore data bytes.

There is no existing mechanism to detect lost or erroneous bits for a transferred file; this type of error is managed at TCP level. The FTP server does not implement a recovery procedure following an error. If transfer is interrupted, the client must resend the complete file.

IMPORTANT: The aim of this section is to define the specificities of the FTP as part of the LIS – BCI interface. It does not provide information on the FTP. For further details on the FTP, refer to the standard RFC0959.

Managing FTP Sessions

Opening a Session

The FTP connects to the FTP client BCI server using an identifier and a password which are recognized (i.e. defined as an active FTP account). Refer to the BCI Link User Manual for more detailed information.

A session can be opened by sending the command:

```
Connecting to the < FTP server>
          < 220 Service ready for new user
     1.   USER <user name>
          < 331 User name okay, need password
     2.   PASS <user password>
          < 230 User logged in, proceed
```

Note: Only valid FTP accounts are authorized to connect to BCI Link, anonymous FTP accounts do not exist.

The following table describes the command sequence that an FTP client should use to set up a new FTP session.

#	Command	Setting	Return Code
1	TCP/IP Connection	• IP address or computer name • Port number (21 by default)	• 220: connection established • 221: connection error • 421: connection refused - too many open sessions (max=10) or unexpected error
2	USER	• FTP account identifier	• 331: identifier OK, need password • 221 or 421: unexpected error
3	PASS	• FTP account password	• 230: session created • 530: incorrect identifier or password • 221 or 421: unexpected error

Configuring Connections for Data Transfer

Type of transfer:

To optimize transfer time and avoid encoding problems, it is preferable to use binary transfers rather than ASCII transfers by sending the following command to the FTP server after connection: `TYPE I.`

If this is successful, the return code is 200. If an error occurs, the return code is 501. This configuration is valid for the duration of the session or until the command `TYPE A` is sent to return to ASCII mode.

Transfer mode (active / passive):

The BCI allows connections in active or passive mode for the transfer of files.

Active mode

Opening of the data connection is initiated by the BCI Link server using the parameters of the `PORT` command previously transmitted by the client.

This command has the following parameters: h1, h2, h3, h4, p1, p2. The first 4 parameters represent the Internet address and the last 2 represent the port for a byte in ASCII representation (refer to the RFC0959 standard).

Note: The server does not implement a default port for the data connection. The client has to send the PORT command beforehand.

Passive mode

Opening of the data connection is initiated by the client for data transfer. To initiate transfer in passive mode, the FTP client must send the following command to the FTP server: PASV. The server will then return the IP address and the port number on which the client should establish the connection for data transfer.

Note: By default, the server runs in active mode which may cause problems with systems which are behind firewalls. Only use passive mode for configurations with a firewall.

The PASV command is only valid for the next data transfer; this is why it must systematically be sent before each data transfer.

Shutting down a Session

The FTP server does not manage session "time outs". In theory, an FTP session can remain open indefinitely. In practice, it is recommended to have a "permanent" session for downloading results (see section about Managing Results (BCI to LIS) below.

A session can be shut down by sending the QUIT command to the FTP server.

Note: Shutting down a session enables only the control connection to be shut down; transfer connection shutdown is initiated by the server following data transfer.

Managing Results (BCI to LIS)

Description of the "RETR" Command

The results are available in the "\upload" directory on the FTP server. Any attempt to upload from a directory other than the "upload" directory is refused by the server.

To upload the contents of a file to the server's current directory, use the command FTP: RETR <File name>. Following a "RETR" command, the FTP server may send one of the following return codes:

Code	Message	Explanation
550	*Requested action not taken,* ***not a file***	This error occurs when you try to upload a file which has the same name as a directory on the server, e.g. **RETR /upload**
550	*Requested action not taken,* ***no access***	This error occurs when you try to upload a file from a directory other than the "upload" directory, e.g. **RETR/download/ monFichier.**

550	*Requested action not taken, **file not found***	This error occurs when you try to upload a file which cannot be found on the server.
451	*Requested action aborted*: **Illegal upload** - should uploaded files having the following definition:	This error occurs when you try to upload a file whose name does not match the naming rules, as defined in ***BCI***.
421	*Service not available, closing control connection*	This error occurs in cases where: • connection for data transfer failed (in active or passive mode). Check that the parameters defined during creation of the connection (Internet address and port number) are correct. • connection for data transfer is suddenly cut off (power cut, connection shut down by the client, etc.).
150	*Opening data connection - transfer mode: <Binary mode*	This message is sent to the client before transfer begins. It indicates that connection has been correctly established and that the transfer mode will be binary or ASCII.
226	*Transfer complete*	The server sends this code to indicate that transfer is complete and that the connection will soon be shut down.

Examples of how the "RETR" Command is used

Example 1:

The FTP command sequence below enables the following actions to occur in succession:

- Establish an FTP session in active mode
- Download a result, following the ***BCI*** rules for naming files
- Shut down the session

```
    Connecting to < FTP server>
            < 220 Service ready for new user
       USER <user name>
            < 331 User name okay, need password
       PASS <user password>
            < 230 User logged in, proceed
       CWD /upload
            < 250 Requested file action okay, completed
       TYPE I
            < 200 Command okay
       PORT h1,h2,h3,h4,p1,p2
            < 200 Command okay
       RETR system1_030909_153020_0.xml
            < 150 Opening data connection - transfer mode: Binary mode
            < 226 Transfer complete
       QUIT
                 < 221 Service closing control connection
```

Example 2:

The FTP command sequence below enables the following actions to occur in succession:

- Establish an FTP session in passive mode
- Upload a result, following the BCI rules for naming files, with the complete path
- Shut down the session

```
Connecting to < FTP server>
            < 220 Service ready for new user
             USER <user name>
            < 331 User name okay, need password
             PASS <user password>
            < 230 User logged in, proceed
             TYPE I
            < 200 Command okay
             PASV
            < 227 Entering Passive Mode (h1,h2,h3,h4,p1,p2)
             RETR /upload/system1_030909_153020_0.xml
            < 150 Opening data connection - transfer mode: Binary mode
            < 226 Transfer complete QUIT
            < 221 Service closing control connection
```

Algorithm for Receipt of Results

The following algorithm describes the main steps for managing the receipt of results. This algorithm is written in pseudocode, i.e. the algorithm gives a description of the tasks to be performed but without language syntax. Refer to the section describing the Conventions to Follow in Writing Pseudocode on page 19.

The principle of this algorithm consists in opening a session, periodically scanning the "upload" directory (every 10 to 30 seconds) and uploading new results.

```
endFtpSession();
// Opening an FTP connection openFTPConnection(ftpHostname, portNumber);
// Opening a new session
startFTPSession(login, pwd);
// Transfer: binary mode setBinaryMode();
LOOP {
// Scanning file names in the upload directory
// Send FTP command : NLST /upload
String[] fileNames = getResultFileNames();
// Transfer all the available files FOR i=0 to fileNames.length() {
// transferring the file using the RETR upload/filename[i] command
getFile(fileNames[i]);
} // end FOR
// Waiting for a time period (in seconds) shorter than the time required
for // scanning the FTP account. This period depends on the BCI Link
//configuration
// it is recommended to set the frequency value between 10 and 30
wait(30);
} // end LOOP
// send QUIT command endFtpSession();
```

FTP Commands supported by the Server

The following table lists all the FTP commands supported by the server.

FTP Command	Description
CWD	**CHANGE WORKING DIRECTORY** This command allows the user to change the working directory of the remote system (retrieval or downloading of files) without altering the current session parameters. The transfer parameters also remain unchanged. The argument is a valid access path in the local file system language.
LIST	**LIST** This command causes the server to send a list. If the pathname specifies a directory or another group of files, the server should transfer a list of files in the specified directory or group. If the pathname specifies a file, then the server should send current information on the file. A null argument implies the user's current working directory. The data transfer is over the data connection in ASCII type. These data may be hard to use automatically in a program, but may be useful to a human user.
NLST	**NAME LIST** This command causes a directory listing to be sent from server to user site. The pathname should specify a directory or other system-specific file group descriptor; a null argument implies the current directory. The server will return a stream of file names and no other information. The data will be transferred in ASCII type over the data connection as valid pathname strings separated by <CRLF> or <NL>. This command is intended to return information that can be used by a program to further process the files automatically.
NOOP	**NO ACTION** It simply prompts the server to send an "OK" reply. This command is useful for checking that the server is operating correctly.

PASS	**PASSWORD** The argument field is a Telnet string specifying the user’s password. This command must immediately be preceded by the USER command, and completes the user’s identification for access control. Since the password is "sensitive" information, it is preferable to "mask" it as the user types it, and to avoid it being displayed on the screen. However, it seems that the server has no means of preventing the password from being disclosed. It is therefore the responsibility of FTP clients to avoid explicit storage of the password and its display.
PASV	**PASSIVE** This command requests the server to "listen" on a data port and to wait for a connection rather than initiate one upon receipt of a transfer command. The response to this command includes the host and port address this server is listening to.
PORT	**DATA PORT** The argument is a host port specification for the data port to be used in data connection. This command and its associated responses are obligatory before transfer in active mode. The argument is the concatenation of a fully qualified TCP/IP address, i.e. a 32-bit Internet address and a 16-bit TCP port address. This address information is divided into 8-bit fields, and the value of each field is transmitted as a decimal number (in character string representation). The fields are separated by commas. A PORT command is thus of the general form: PORT h1,h2,h3,h4,p1,p2 where h1 is the high order 8 bits of the Internet host address.
PWD	**PRINT WORKING DIRECTORY** This command causes the name of the current working directory to be returned in the reply.
XPWD	Identical to the PWD command.
QUIT	**LOGOUT** This command terminates a USER session and closes the control connection.

RETR	**RETRIEVE** This command causes the server to transfer a copy of the file, specified in the pathname, to the server at the other end of the data connection. The status and contents of the file at the server site will be unaffected.
STOR	**STORE** This command causes the server to accept the data transferred via the data connection and to store the data as a file at the server site. If the file specified in the pathname exists at the server site, then its contents will be replaced by the data being transferred. A new file is created at the server site if the file specified in the pathname does not already exist.
SYST	**SYSTEM** This command is used to return the type of operating system which hosts the FTP server. Whatever the OS, this command returns a single "UNIX" value.
TYPE	**TYPE** This argument specifies the type of representation for data used. This command manages a single parameter: I : binary transfer A : ASCII transfer
USER	**USER NAME** This argument field is a Telnet string identifying the user. The user identifier is that which is required by the server for access to the file system of the server host. This command will normally be the first command transmitted by the user after the control connections are made. The password is also required by the server. The server will accept a new USER command at any time. This cancels any reference to the user, the password and the previous account by restarting the session opening sequence. All the transfer parameters remain unchanged and any file transfer in progress will be completed with the old session parameters.
HELP	**HELP** Returns the list of commands accepted by the server.

IMPORTANT: A connection to an FTP server for transferring data must be done with the use of a client. The client may seemingly use a set of commands which differs from the server's ones; these commands are Command Aliases, a shorthand for actual commands.

These command aliases vary from one client to another.

Here is a list of common Command aliases contained in many Clients and their equivalents:

Alias	Equivalent	Alias	Equivalent
!	QUIT	**mget**	RETR (used to automate the retrieval of multiple files)
ascii	TYPE A	**mls**	LIST (list contents of multiple directories)
binary	TYPE I	**mdir**	LIST (list contents of multiple directories)
bye	QUIT	**mput**	STOR (Stores multiple files)
cd	CWD	**put**	STOR
dir	LIST	**pwd**	PWD
get	RETR	**quit**	QUIT
ls	NLIST	**recv**	RETR
send	STOR	**type**	TYPE
remotehelp	HELP	**user**	USER

FTP Return Codes

Below is a list of the FTP return codes with an explanation of what each means.

Code	Explanation
150	File status okay; about to open data connection.
200	Command okay.
202	Command not implemented, superfluous at this site.
215	Name system type. Where name is an official system name from the list in the Assigned Numbers document.
220	Service ready for new user.
221	Service closing control connection.
226	Closing data connection. Requested file action successful (for example, file transfer or abort).

227	Entering Passive Mode (h1,h2,h3,h4,p1,p2).
230	User logged in.
250	Requested file action completed.
257	"PATHNAME" created.
331	User name okay, need password.
421	Service not available, closing control connection. This may be a reply to any command if the service knows it must shut down.
425	Can’t open data connection.
451	Requested action aborted. Local error in processing.
500	Syntax error, command unrecognized. This may include errors such as command line too long.
501	Syntax error in parameters or arguments.
530	Not logged in.
550	Requested action not taken. File unavailable (e.g., file not found, no access).

Conventions to Follow in Writing Pseudocode

Below is a list of the conventions to follow.

Characters / Keywords	Explanation
LOOP	Describes an infinite loop, the code executed in the loop is enclosed in brackets.
FOR var=MinValue to MaxValue	Iteration on a variable `var` initialized at a minimum value (`MinValue`) and iterated until the `MaxValue` (`MaxValue exclusive`) is reached. The code to be executed for each iteration is enclosed in brackets.
IF condition THEN..ELSE	The code enclosed in brackets which follows the `THEN` clause is `only` executed if the condition is checked. If the condition is not checked and an `ELSE` clause exists, the code enclosed in brackets which follows the `ELSE` clause, is executed.
var = value	Assigns a value "`value`" to a variable "`var`".
A == B	Tests equality between two variables A and B.
// comment	A green line preceded by the character "//" represents a comment.
Return = function (argument1, argument2, ..)	The functions that perform operations can be defined with or without arguments and can return a value. For functions without an argument, the syntax is MyFunction().
object.function(argument1, ..)	Syntax applied to object-oriented languages. The function "`function`" is applied to the object "`object`".

CHAPTER 4: Shared Folder Protocol

FilmArray® can transmit the test results using the Shared Folder protocol, as described in this section, or the File Transfer Protocol, as described in File Transfer Protocol on page 9.

Analysis Result Naming Rules

Analysis result files sent by the AI is named as follows:

```
FILMARRAY_YYMMDD_HHMMSS_s.xml
```

Where:

- **YYMMDD** means Year, Month, Day.
- **HHMMSS** means Hour, Minute, Second.
- **s** is the sequence between 0 and 99. This is used if the AI sends more than one analysis result (in the same second).

Shared Folder Protocol Description

This protocol enables the LIS to communicate with BCI Link through files put in shared folders.

The shared folders can be located on the same system as BCI Link or on a remote system.

BCI Link must have the read and write access rights on the shared folder (local or remote).

The analysis request must be copied in the request shared folder defined in BCI Link. The file name must be compliant with the Analysis Result Naming Rules.

In order to avoid concurrent access to this file, the file must be copied in the folder with a different extension than ".xml" then renamed after the copy with the extension ".xml."

BCI Link puts the results in a "result" folder with a given extension. Optionally, BCI Link can generate warnings if a file is pending for too much time in the result folder.

The analysis result must be remove from the result shared folder defined in BCI Link. BCI Link will automatically remove it if the file is only copied and not cut from the folder.

Appendix - Glossary

AI

Analysis instrument or analysis system. BioFire Diagnostics LLC AI = FilmArray 2.0 system Computer.

Analysis request

A message produced by the LIS containing one or more analysis requests. An analysis request consists of one or more tests for a specific type of analysis. These analysis requests can be attributed to one or more AIs.

ASCII

American Standard Code for Information Interchange.

ASTM

American Society for Testing and Materials.

ASTM E1394

Standard specification for transferring information between clinical instruments and computer systems, published in the early 1990's by the E31 committee and the American Society for Testing and Materials.

📖 Reference Web site: http://www.nccls.org

ASTM-XML representation

Specifications of messages relating to the transfer of data between the AI and the LIS. This specification, elaborated by bioMérieux, is based on the ASTM 1394 attributes with an XML message structure.

Attribute

Characteristic element.

BCI

bioMérieux Communication Interface.

BMX

Abbreviation for bioMérieux.

Data representation

Type of description used to structure messages for data transfer between analysis instruments and the laboratory information systems.

Field

A field is a discrete unit of information such as a patient name, sex, primary physician, etc. Each field is preceded by a code that describes the type of field and also implies its length. All fields end with one to three user-defined terminating characters (chosen from a subset of ASCII characters). Each field has a maximum length in characters. BCI Link will accept data that are less than or equal to the maximum number of allowable characters. Most fields that are longer than the number of significant characters are truncated and the text is left-justified.

Note: The only exception is the exam ID (ci), which is truncated and right justified.

FTP

The FTP service is based on a client / server, architecture and is used to transfer files between heterogeneous machines. It is based on the TCP/IP network protocols.

For further information, refer to the standard RFC 0959 - http://www.rfcs.org.

LIS

Laboratory Information System.

N/A

Not Applicable

Results message

A message produced by an AI which attaches information to the analysis requests contained in the analysis request. This information includes all the useful or essential data corresponding to the results of the analyses.

Session

A session consists of a connection between the host and the FilmArray 2.0 system Computer. Data transfer is unidirectional from the computer to the host. The initiator of the session is the sender and the other computer is the receiver. The ASCII line request character **<ENQ>** starts the session. **<EOT>** signals the end of the message and session.

Socket

Communication connection.

Specimen ID

Unique identifier used by the lab to identify a clinical specimen.

TBD

To Be Defined.

TCP/IP

Transport Control Protocol / Internet Protocol.

Test order

A series of tests to be performed for a sample. An analysis request consists of a series of test orders

Type of test

Type of analysis that an AI is capable of performing on a sample using a reagent kit and an analysis procedure.

XML

Abbreviation of "Extensible Markup Language". XML is a markup specification language for structuring documents.

📖 Reference Web site: http://www.w3.org/XML/

XML schema

A method for describing XML document, mainly including a description of all the data types (XML element and attribute) and the document structure. Tools are used to validate an XML source with an XML schema.

📖 Reference Web site: http://www.w3.org/XML/Schema.Revision History

For additional information regarding FilmArray LIS interface driver development, e-mail LISSupport@biofiredx.com.

BIOFIRE®

A BIOMÉRIEUX COMPANY

For additional information regarding our products and applications, please contact BioFire Diagnostics Customer Support Department, local sales representative or an authorized distributor.