2.3.3.3. Details of binary sections

All binary sections are contained within semicolon-delimited text fields, which may be thought of as 'binary strings'. Each such binary string must be given as a value of an appropriate CIF tag (usually `_array_data.data`), either directly or within a loop.

Before getting to the actual binary data, there are some preliminaries to allow a smooth transition from the conventions of CIF to those of raw streams of 'octets' (8-bit bytes). Within the binary-data text string, the conventions developed for transmitting e-mail messages including binary attachments are followed. There is secondary ASCII header information, formatted as MIME headers [see RFCs 2045–2049 by Freed & Borenstein (1996*a*,*b*,*c*), Freed *et al.* (1996) and Moore (1996)]. The boundary marker for the beginning of all this is the special string `--CIF-BINARY-FORMAT-SECTION--` at the beginning of a line. The initial '`--`' says that this is a MIME boundary. We cannot put '`###`' in front of it and conform to MIME conventions.

Immediately after the boundary marker are MIME headers, describing some useful information we will need to process the binary section. Only a few headers with a narrow range of values have to be understood to process a CBF (as opposed to an imgCIF, for which the headers can be more varied).

The 'Content-Type' header may be any of the discrete types permitted in RFC 2045 (Freed & Borenstein, 1996*b*); 'application/octet-stream' is recommended. If an octet stream was compressed, the compression should be specified by the parameter `conversions="x-CBF_PACKED"` or by the parameter `conversions="x-CBF_CANONICAL"`. Other compression schemes may be added at a future date. Details of the compression schemes are given in Chapter 5.6.

The 'Content-Transfer-Encoding' header should be `BINARY` for a CBF. For an ASCII imgCIF file, the 'Content-Transfer-Encoding' header may be `BASE64`, `QUOTED-PRINTABLE`, `X-BASE8`, `X-BASE10` or `X-BASE16`. The `BASE64` encoding is a reasonably efficient encoding of octets (approximately eight bits for every six bits of data) and is recommended for imgCIF files. The octal, decimal and hexadecimal transfer encodings are for convenience in debugging and are not recommended for archiving and data interchange.

The 'X-Binary-Size' header specifies the size of the binary data in octets. If compression was used, this size is the size after compression, including any book-keeping fields. In general, no portion of the binary header is included in the calculation of the size.

The 'X-Binary-Element-Type' header specifies the type of binary data in the octets, using the same descriptive phrases as in `_array_structure.encoding_type`. The full list of valid types is:

```
unsigned 8-bit integer
signed 8-bit integer
unsigned 16-bit integer
signed 16-bit integer
unsigned 32-bit integer
signed 32-bit integer
signed 32-bit real IEEE
signed 64-bit real IEEE
signed 32-bit complex IEEE
```

The default value is `unsigned 32-bit integer`. See IEEE (1985) for details on the IEEE formats.

The MIME header items are followed by an empty line and then by a special sequence marked here as `START_OF_BIN`, consisting of Ctrl-L, Ctrl-Z, Ctrl-D and a single binary flag character of hexadecimal value D5 (213 decimal). The binary data follow immediately after this flag character. The control (Ctrl-. . .) characters are inserted as a convenience to inhibit accidental listing or printing of large amounts of binary data on character-oriented output devices such as terminals or printers.

In general, if the value given for 'Content-Transfer-Encoding' is one of the real encodings `BASE64`, `QUOTED-PRINTABLE`, `X-BASE8`, `X-BASE10` or `X-BASE16`, this file is an imgCIF. More details on the imgCIF encodings are given in Section 2.3.5 below.

For either a CBF or an imgCIF, the optional 'Content-MD5' header provides a sophisticated check [the 'RSA Data Security, Inc. MD5 Message-Digest Algorithm' (Rivest, 1992)] on the integrity of the binary data.

In a CBF, the raw binary data begin after an empty line terminating the MIME headers and after the `START_OF_BIN` identifier. `START_OF_BIN` contains bytes to separate the ASCII lines from the binary data, bytes to try to stop the listing of the header, bytes which define the binary identifier, which should match the `*.binary_id` defined in the header, and bytes which define the length of the binary section.

Octet	Hexadecimal	Decimal	Purpose
1	0C	12 (Ctrl-L)	End the current page
2	1A	26 (Ctrl-Z)	Stop listings in MS-DOS
3	04	04 (Ctrl-D)	Stop listings in Unix
4	D5	213	Binary section begins
$5 \ldots 5+n-1$			Binary data (n octets)

Only bytes $5 \ldots 5+n-1$ are encoded for an imgCIF file using the indicated Content-Transfer-Encoding.

The binary characters serve specific purposes:

(i) The Ctrl-L will terminate the current page in listings in most operating systems.

(ii) The Ctrl-Z will stop the listing of the file in MS-DOS-type operating systems.

(iii) The Ctrl-D will stop the listing of the file in Unix-type operating systems.

(iv) The unsigned byte value 213 (decimal) is binary 11010101 (octal 325, hexadecimal D5). This has the eighth bit set, so it can be used to detect the error of seven-bit rather than eight-bit transmission. It is also asymmetric, providing one last check for reversal of bit order within bytes.

The carriage return, line feed pair before the `START_OF_BIN` and other lines can also be used to check that the file has not been corrupted (*e.g.* by being sent by ftp in ASCII mode).

The 'line separator' immediately precedes the 'start of binary identifier', but blank spaces may be added prior to the preceding 'line separator' if desired (*e.g.* to force word or block alignment). The binary data do not have to completely fill the bytes defined by the byte-length value, but clearly cannot be greater than this value (except when the value zero has been stored, which means that the size is unknown, and no other headers follow). The values of any unused bytes are undefined. The end of binary section identifier is placed exactly at the byte following the full binary section as defined by the length value. The end of binary section identifier consists of the carriage return/line feed pair followed by

```
--CIF-BINARY-FORMAT-SECTION----
;
```

with each of these lines followed by the carriage return/line feed pair. This brings us back into a normal CIF environment.

The first 'line separator' separates the binary data from the pseudo-ASCII line. The end-of-binary-section identifier is in a sense redundant, since the binary-data-length value tells a program how many bytes to jump over to the end of the binary data. However, this redundancy has been deliberately added for error checking, and for possible file recovery in the case of a corrupted file. This identifier must be present at the end of every block of binary data.