Chapter 2
Control/Query

2.1 INTRODUCTION

The Control/Query chapter of this Standard defines the generic rules that apply to all messages. Subsequent sections define functionally specific messages to be exchanged among certain applications. The specific aspects of message definition that are addressed herein are:

a)the form to be used in functional chapters for describing messages. This includes their purpose, their contents, and the interrelationships among them. This form is called an **abstract message definition** because it is purely a level 7 (application) definition.

b) the HL7 encoding rules for converting an abstract message into a string of characters that comprise an actual message

c)the programming procedures required to exchange messages using the HL7 specifications

d)the anticipated relationship with lower level protocols

e)certain message segments that are components of all messages

f)a single message, the Acknowledgement message, that may be used unchanged in multiple applications

2.3 CONCEPTUAL APPROACH

2.4.1 Trigger events

The Standard is written from the assumption that an event in the real world of healthcare creates the need for data to flow among systems. The real-world event is called the **trigger event**. For example, the trigger event **a patient is admitted** may cause the need for data about that patient to be sent to a number of other systems. The trigger event, **an item is used from floor stock on behalf of a patient**, may cause the need for information about the patient and the item used to be sent from the patient care system to the patient accounting system and materials management system. When the transfer of information is initiated by the application system that deals with the triggering event, the transaction is termed an **unsolicited update**.

Note: No assumption is made about the design or architecture of the application system creating the unsolicited update. The scope of HL7 is restricted to the specification of messages between application systems, and the events triggering them.

HL7 allows the use of trigger events at several different levels of data granularity and inter-relationship. For example, most ADT trigger events concern single objects (such as an admit event, which creates a message that contains data about a single person and/or account). Other ADT trigger events are concerned with relationships between more than one object (e.g., the merge events, which specify patient or account merges). Some ADT trigger events pertain to a collection of objects that may have no significant interrelationships (e.g., a record-oriented location-based query, whose response contains data about a collection of inpatients who are related only temporarily by local geography).

2.4.3 Acknowledgements: original mode

When the unsolicited update is sent from one system to another, this acknowledgement mode specifies that it be acknowledged at the application level. The reasoning is that it is not sufficient to know that the underlying communications system guaranteed delivery of the message. It is also necessary to know that the receiving application processed the data successfully at a logical application level.

The acknowledgement may contain data of interest to the system that initiated the exchange. For example, if a patient care system has processed the trigger event **a lab test is ordered for a patient**, it may send an unsolicited update to a lab application identifying the patient, the test ordered, and various other information about the order. The ancillary will acknowledge the order when it has processed it successfully. For some pairings of patient care and ancillary department systems the acknowledgement may also include the ancillary identification number that was assigned. (HL7 does not require Order Entry and Results Reporting applications to interface in this manner, but it supports those that do.)

The HL7 Standard makes no assumptions about the ownership of data. It also makes no requirements of its own on the subsequent action of the recipient of data, nor does it make any assumption about the design or architecture of the receiving application system. The scope of HL7 is restricted to the specification of messages between application systems, and the events triggering them. HL7 does not explicitly support, but can be used with, systems that support store and forward and data broadcast facilities (see the HL7 Implementation Guide).

The HL7 Standard makes no functional interpretation of the requirement that a system commit the data in a message to its database before acknowledging it. All that is required is that the receiving system accept responsibility for the data, providing the same integrity test that it would apply to data from any source. To continue the prior example, the ancillary system may acknowledge the order after placing it in an input queue, expecting to fully process the order into its database at a future time. The only assumption is that the input queue is maintained at the same level of integrity as the database.

2.4.5 Acknowledgements: enhanced mode

The HL7 acknowledgement paradigm has been extended to distinguish both accept and application acknowledgements, as well the conditions under which each is required. With a positive accept acknowledgement, the receiving system commits the message to safe storage in a manner that releases the sending system from the need to resend the message. After the message has been processed by the receiving system, an application acknowledgement may be used to return the resultant status to the sending system.

2.4.7 Queries

A different data exchange occurs when one system sends a query to another. For example, in a cardiac catheterization application, there may be a trigger event **a procedure is scheduled** for a patient who is not already registered in the cardiac catheterization application's database. The application may send a request message containing the patient's ID number to the ADT system and receive a response containing the necessary data to permit processing of the order. This requesting transaction is a **query**, as distinguished from the unsolicited update discussed above. The information that flows between the systems is contained in the response. The response itself is not acknowledged with a third message.

In all cases, the HL7 Standard consists of a simple exchange of messages between a pair of applications: the unsolicited update and its acknowledgement or the query and its response. The underlying operational model is that of a **client** and a **server**. An application interfaces with another application using an event code that identifies the transaction. The other application responds with a message that includes data or an error indication. The initiating application may receive a reject status from the other application or from lower level software indicating that its message was not received correctly.

2.5 COMMUNICATIONS ENVIRONMENT

The HL7 Standard defines the messages as they are exchanged among applications entities and the procedures used to exchange them. As such, it conceptually operates at the seventh level of the ISO model for Open System Interconnection (OSI). It is primarily concerned with the data content and interrelationship of messages and with communicating certain application level error conditions.

Since the OSI protocols are not universally implemented, the HL7 Working Group is interested in providing standards that will be useful in the interim. It is also recognized that there is now, and will continue to be, interest in communicating health data among systems operating in communications environments that provide a high level of functionality, but use protocols other than ISO OSI . The universe of environments of interest to HL7 include, but is not restricted to:

a)ad hoc environments that do not provide even basic transport reliability. Such environments consist of point-to-point RS-232 links, modems, and even LANs, if their connection to host computers is made via RS-232 communications links. Until OSI high level standards become truly prevalent, many healthcare interfaces will be implemented over such links. In such an environment, the HL7 Lower Level Protocols (LLP) may be used between systems to enhance the capabilities of the communications environment. The HL7 Lower Level Protocols are defined in HL7 Implementation Guide, which is not an official part of the standard.

b)environments that support a robust transport level, but do not meet the high level requirements. This includes environments such as TCP/IP, DECNET, and SNA.

c)ISO and proprietary networks that implement up to presentation and other high level services. IBM's SNA LU6.2 and SUN Microsystems's NFS are examples of complete proprietary networks.

d)two or more applications running on the same physical and/or logical machine that are not tightly integrated. In these environments, the messaging capabilities may be provided by inter-process communications services (e.g., Pipes in a UNIX System).

The HL7 standard assumes the communications environment will provide the following capabilities:

a)error free transmission. Applications can assume that they correctly received all of the transmitted bytes in the correct order that they were sent. This implies that error checking is done at a lower level. However, sending applications may not assume that the message was actually received without receiving an acknowledgement message.

b)character conversion. If the two machines exchanging data use different character sets, the communications environment will convert the data from one set to the other.

c)message length. HL7 sets no limits on the maximum size of HL7 messages. The Standard assumes that the communications environment can transport messages of any length that might be necessary. In practice, sites may agree to place some upper bound on the size of messages and may use the message continuation protocol, described later in this chapter, for messages that exceed the upper limit.

Note: Just as HL7 makes no assumptions about the design or architecture of the application systems sending and receiving HL7 messages, it makes no assumptions about the communications environment beyond those listed above. In particular, aside from the above assumptions, the communications environment, including its architecture, design and implementation, is outside the scope of HL7.

2.7 HL7 MESSAGES

This section defines the components of messages and provides the methodology for defining abstract messages that is used in later chapters.

2.8.1 Message definition

A **message** is the atomic unit of data transferred between systems. It is comprised of a group of segments in a defined sequence. Each message has a **message type** that defines its purpose. For example the ADT Message type is used to transmit portions of a patient's ADT data from one system to another. A three character code contained within each message identifies its type. These are listed in the Message Type list, Appendix A.

The real-world event that initiates an exchange of messages is called a trigger event. (See section 2.2.1 for a more detailed description of trigger events.) Appendix A contains the codes that represent all defined trigger events. These codes represent values such as **A Patient is Admitted** or **An order event occurred**. There is a one-to-many relationship between message types and trigger event codes. The same trigger event code may not be associated with more than one message type; however a message type may be associated with more than one trigger event.

All message type and trigger event codes beginning with Z are reserved for locally defined messages. No such codes will be defined within the HL7 Standard.

2.8.3 Segments

A **segment** is a logical grouping of **data fields**. Segments of a message may be required or optional. They may occur only once in a message or they may be allowed to repeat. Each segment is given a name. For example, the ADT message may contain the following segments: Message Header (MSH), Event Type (EVN), Patient ID (PID), and Patient Visit (PV1).

Each segment is identified by a unique three character code known as the Segment ID. Although the actual segments are defined in various chapters the ID codes assigned to all segments are listed in Appendix A.

All segment ID codes beginning with Z are reserved for locally defined messages. No such codes will be defined within the HL7 Standard.

2.8.5 Fields

A field is a string of characters. HL7 does not care how systems actually store data within an application. When fields are transmitted, they are sent as character strings. Except where noted, HL7 data fields may take on the null value. Sending the null value, which is transmitted as two double quote marks (""), is different from omitting an optional data field. The difference appears when the contents of a message will be used to update a record in a database rather than create a new one. If no value is sent, (i.e., it is omitted) the old value should remain unchanged. If the null value is sent, the old value should be changed to null.

The various chapters of the Standard contain segment definition tables. These tables list and describe the data fields in the segment and characteristics of their usage. A comprehensive data dictionary of all HL7 fields is provided in Appendix A. In defining a segment, the following information is specified about each field:

2.8.6.1**position:** Ordinal position of the data field within the segment. This number is used to refer to the data field in the text comments that follow the segment definition table.

2.8.6.2**name:** Globally unique descriptive name for the field.

2.8.6.3**ID number:** Small integer that uniquely identifies the data field throughout the Standard. The ID number is not significant under the HL7 message encoding rules but is included as a convenience for those who would apply the HL7 Standard using other message encoding rules.

2.8.6.4**maximum length:** Maximum number of characters that one occurrence of the data field may occupy. The maximum length is not of conceptual importance in the abstract message or the HL7 coding rules. It is included because it helps readers understand the purpose of the field and it may have pragmatic importance in specific implementations. It is calculated to include the component and subcomponent separators that are defined below. Because the maximum length is that of a single occurrence, the repetition separator is not included in calculating the maximum length. (See Section 2.4.3.6)

2.8.6.5**optionality:** Whether the data field is required, optional, or conditional in a segment. The designations are:

R	-	required
O	-	optional

C-conditional on the trigger event

2.8.6.6**repetition:** Whether the field may repeat. The designations are:

N	-no repetition
Y	-the field may repeat an indefinite or site determined number of times

(integer) -the field may repeat up to the number of times specified in the integer

Each occurence may contain the number of characters specified by the field's maximum length. (See Section 2.4.3.4)

2.8.6.7**table:** HL7 defines a table of values for this field. An entry in the table number column means that the table name and the element name are equivalent.

The manner in which HL7 defines the valid values for tables will vary. Certain fields, like Patient Location, will have values that vary from institution to institution. Such tables are designated user or site-defined. Even though these tables are not defined in the standard, they are given an HL7 table number to facilitate implementations. The ID data type is often used to encode values for these tables. Note that some of these tables (e.g., location) may reference common master files.

Others, like Event Type (table 0003), are a part of the HL7 Standard because they affect the interpretation of the messages that contain them. They are limited to the values established by the HL7 Standard. The ID data type is most often used to encode values for HL7 tables. When an HL7 table exists it is strongly recommended that it be used. The values are listed in Appendix A. These HL7 tables also appear in the text in a standard box format (e.g., the Event Type table in section 3.3.1.1). Additions may be included on a site-specific basis.

Still other tables contain values that are encoded by reference to other standards documents. For example, the encoding for Lab procedures is defined by ASTM 1238-88. The CE data type is used to encode values for these tables.

Finally, there are some user-defined tables that contain values that might be standardized across institutions but for which no applicable official standard exists. For these a set of **suggested** values may be listed in Appendix A. These suggested values appear in the text in a standard non-box format (e.g., the Event Reason Code -- table 0062 in section 3.3.1.4). It is expected that these values will be used where applicable within an institution and serve as a basis for extensions as required. The appropriate functional committee within HL7 solicits suggestions for additional values from institutions that are applying the Standard.

Various HL7 data types (ID, CE, CF, CK, CM, CN, CQ, and RP) are used to convey tabular values, or have a component containing tabular values).

2.8.6.8**data type:** Restrictions on the contents of the data field. There are a number of data types defined by HL7. These will be explained in Section 2.4.5-data types.

2.8.7 Message delimiters

In constructing a message certain special characters are used. They are the segment terminator, the field separator, the component separator, subcomponent separator, repetition separator, and escape character. The segment terminator is always a carriage return (in ASCII, a hex 0D). The other delimiters are defined in the MSH record, with the field delimiter in the 4th character position, and the other delimiters occurring as in the field called Encoding Characters, which is the first field after the segment ID. The delimiter values used in the MSH segment are the delimiter values used throughout the entire message. In the absence of other considerations, HL7 recommends the suggested values found in *figure 2-1*.

At any given site, the subset of the possible delimiters may be limited by negotiations between applications. This implies that the receiving applications will use the agreed upon delimiters, as they appear in the Message Header segment (MSH), to parse the message.

Figure 2-1 Delimiter Values

Delimiter	Suggested Value	Encoding Character Position	Usage
Segment Terminator	<cr> hex 0D	-	Terminates a segment record. This value cannot be changed by implementors.
Field Separator	|	-	Separates two adjacent data fields within a segment. It also separates the segment ID from the first data field in each segment.
Component Separator	^	1	Separates adjacent components of data fields where allowed.
Sub-Component Separator	&	4	Separates adjacent subcomponents of data fields where allowed. If there are no subcomponents, this character may be omitted.
Repetition Separator	~	2	Separates multiple occurrences of a field where allowed.
Escape Character	\	3	Escape character for TX and FT fields. If no escape characters are used in a message, this character may be omitted. However, it must be present if subcomponents are used in the message.

2.8.9 Data types

2.8.10.1 ST string data

String data is left justified with trailing blanks optional. Any displayable (printable) ACSII characters (hexadecimal values between 20 and 7E, inclusive). Example:

```
|almost any data at all|
```

To include any HL7 delimiter character (except the segment terminator) within a string data field, use the appropriate HL7 escape sequence (see section 2.4.6.1).

2.8.10.2 TX text data

String data meant for user display (on a terminal or printer). Such data would not necessarily be left justified since leading spaces may contribute greatly to the clarity of the presentation to the user. Because this type of data is intended for display, it may contain certain escape character sequences designed to control the display. Escape sequence formatting is defined later in this chapter in Section 2.4.6. Leading spaces should be included. Trailing spaces should be removed. Example:

```
| leading spaces are allowed.|
```

Since TX data is intended for display purposes, the repeat delimiter, when used with a TX data field, implies a series of repeating lines to be displayed on a printer or terminal. Therefore, the repeat delimiters are regarded as paragraph terminators or hard carriage returns (e.g., they would display as though a CR/LF were inserted in the text).

A receiving system would word-wrap the text between repeat delimiters in order to fit it into an arbitrarily sized display window but start any line beginning with a repeat delimiter on a new line.

2.8.10.3 FT formatted text data

This data type is derived from the string data type by allowing the addition of embedded formatting instructions. These instructions are limited to those that are intrinsic and independent of the circumstances under which the field is being used. The actual instructions and their representation are described later in this chapter. The differences from a string data field and an FT field is of arbitrary length (up to 65k) and may contain formatting commands enclosed in escape characters. Example:

```
|\.sp\(skip one vertical line)|
```

For additional examples of formatting commands see Section 2.4.6-use of escape sequences in text fields, below.

2.8.10.4 NM numeric

A number represented as a series of ASCII numeric characters consisting of an optional leading sign (+ or -), the digits and an optional decimal point. In the absence of a sign, the number is assumed to be positive. If there is no decimal point the number is assumed to be an integer. Examples:

```
|999|
|-123.792|
```

Leading zeros, or trailing zeros after a decimal point, are not significant. The two values 01.20 and 1.2 are identical. Except for the optional leading sign (+ or -) and the optional decimal point (.), no non-numeric ASCII characters are allowed. Thus, the value <12 should be encoded as a string data type.

2.8.10.5 DT date

Always in the format YYYYMMDD. Example:

```
|19880704|
```

2.8.10.6 TM time

Always in the format HHMM[SS[.SSSS]][+/-ZZZZ] using a 24 hour clock notation. The seconds designation (SS) is optional. If not present it will be interpreted as 00. The fractional seconds designation is likewise optional. If not present it will be interpreted as .0000. The fractional seconds could be sent by a transmitter who requires greater precision than whole seconds. Fractional representations of minutes, hours or other higher orders units of time are not permitted. The time zone of the sender may be sent optionally as an offset from the coordinated universal time (previously known as Greenwich Mean Time.) Where the time zone is not present in a particular TM field but is included as part of the date/time field in the MSH segment, the MSH value will be used as the default time zone. Otherwise, the time is understood to refer to the local time of the sender. Midnight is represented as 0000. Examples:

|235959+1130|1 second before midnight in a time zone eleven and half hours ahead of Universal Coordinated Time (i.e., east of Greenwich).

|0800| Eight AM, local time of the sender.

|093544.2312|44.2312 seconds after Nine thirty-five AM, local time of sender.

2.8.10.7 TS time stamp

Contains the exact time of an event, including the date and time. Time stamp fields are always in the format:

YYYYMMDD[HHMM[SS[.SSSS]]][+/-ZZZZ]^<degree of precision>

The date portion of a time stamp follows the rules of a date field and the time portion follows the rules of a time field. When used as a birthdate, the HHMM portion is optional. If not present the HHMM portion will default to 0000, i.e., midnight of the day just beginning. The specific data representations used in the HL7 encoding rules are compatible with ISO 8824-1987(E). An optional second component indicates the degree of precision of the date (Y = year, L = month, D = day, H = hour, M = minute, S = second). (Maximum length of field is 26). Examples:

|17760704010159-0600|1:01:59 on July 4, 1776 in the Eastern Standard Time zone.

|17760704010159-0500|1:01:59 on July 4, 1776 in the Eastern Daylight Saving Time zone.

|198807050000| Midnight of the night extending from July 4 to July 5, 1988 in the local time zone of the sender.

|198807050000^D| Same as prior example, but precision extends only to the day. Could be used for a birthdate.

The HL7 Standard strongly recommends that all systems routinely send the time zone offset but does not require it. All HL7 systems are required to accept the time zone offset, but its implementation is application specific. For many applications the time of interest is the local time of the sender. For example, an application in the Eastern Standard Time zone receiving notification of an admission that takes place at 11:00 PM in San Francisco on December 11 would prefer to treat the admission as having occurred on December 11 rather than advancing the date to December 12.

One exception to this rule would be a clinical system that processed patient data collected in a clinic and a nearby hospital that happens to be in a different time zone. Such applications may choose to convert the data to a common representation. Similar concerns apply to the transitions to and from daylight saving time. HL7 supports such requirements by requiring that the time zone information be present when the information is sent. It does not, however, specify which of the treatments discussed here will be applied by the receiving system.

2.8.10.8 PN person name

<family name> ^ <given name> ^ <middle initial or name> ^ <suffix (e.g., JR or III)> ^ <prefix (e.g., DR)> ^ <degree (e.g., MD)>

A name includes multiple free text components as listed above. The maximum length of a PN field is 48 characters including component separators. The sending system may send upper- and lowercase or all uppercase. The receiving system may convert to all uppercase if required. Example:

```
|SMITH^JOHN^J^III^DR^PHD|
```

2.8.10.9 TN telephone number

For use in the United States and conforming countries, the telephone number is always in the form:

```
[NN] [(999)]999-9999[X99999][B99999][C any text]
```

The optional first two digits are the country code. The optional **X** portion gives an extension. The optional **B** portion gives a beeper code. The optional **C** portion may be used for comments like, **After 6:00**. While no explicit limit is placed on the text field, receiving systems may be expected to truncate values that are more than 10 characters long. To accommodate the variability of institutional phone systems, the length of the extension and beeper numbers may be extended by local agreement. Examples:

```
|(415)925-0121X305|
|234-4532CWEEKENDS|
```

2.8.10.10 AD address

```
<street address> ^ < other designation> ^ <city> ^ <state or province> ^ <zip or postal code> ^ <country> ^ <type> ^ <other
geographic designation>
```

All components are ST data type. The street or mailing address of a person or institution. For use within North America:

a) state or province should be represented by the official US Postal service two-letter codes
b) zip takes the form 99999[-9999], Canadian postal code is 6 alpha-numeric characters
c) the country code is assumed to be USA if null
d) other geographic designation includes county, bioregion, SMSA, etc.

Type is optional and defined by *table 0190 - address type*.

Example:

```
|10 ASH LN^#3^LIMA^OH^48132^""^|
```

Table 0190 Address type

Value	Description
C	current or temporary
P	permanent
M	mailing
B	business
O	office
H	home

2.8.10.11 ID coded value

The value of such a field follows the formatting rules for an ST field except that it is drawn from a table of legal values. Examples of ID fields include religion and sex.

2.8.10.12 SI sequence ID

A positive integer in the form of an NM field. The uses of this field are defined in the chapters defining the segments and messages in which it appears.

2.8.10.13 CM composite

A field that is a combination of other meaningful data fields. Each portion is called a **component**. The specific components of CM fields are defined within the field descriptions. Certain other composites have been separately identified and are described below. The use of this data type will be slowly phased out and new unique data types will be created.

Wherever a component of an HL7 field is itself an HL7 data type which contains components, its delimiters are demoted by one. Thus a component designated as a CE data type should be encoded as <identifier & text & name of coding system> (see section 2.4.5.17, below). Note that since HL7 delimiters are not recursive, an HL7 data type containing components cannot be a sub-component. When this level of detail is needed, each component of the HL7 data type can be encoded as a separate subcomponent. For an example of this, see the encoding of the filler order number in the order sequencing component of the Timing/Quantity data type, (figure 4-7, section 4.4.10.1, below).

2.8.10.14 CK composite ID with check digit

<ID number (NM)> ^ <check digit (NM)> ^ <code identifying the check digit scheme employed (ID)> ^ <assigning facility ID (ST)>

This data type is used for certain fields that commonly contain check digits, e.g., *PID-3-Patient ID (Internal ID)*. If a site is not using check-digits for a particular CK field, the second and third components are not valued.

The check digit in this data type is not an add-on produced by the message processor. It is the check digit that is part of the identifying number used in the sending application. If the sending application does not include a self-generated check digit in the indentifying number, this component should be valued null.

The assigning facility ID is a unique name (up to six characters in length) of the system that stores the data. It is an ST data type. It is equivalent to the application ID of the placer or filler order number (see Chapter 4). Assigning facility ID's are unique across a given HL7 implementation.

The check digit scheme codes are defined in *table 0061 - check digit scheme*.

Table 0061 Check digit scheme

Value	Description
M10	Mod 10 algorithm
M11	Mod 11 algorithm

Example:

```
|128952^6^M11^ADT01|
```

The algorithm for calculating a Mod10 check digit is as follows:

Assume you have an identifier = 12345. Take the odd digit positions, counting from the right, i.e., 531, multiply this number by 2 to get 1062. Take the even digit positions, starting from the right (i.e., 42), prepend these to the 1062 to get 421062. Add all of these six digits together to get 15. Subtract this number from the next highest multiple of 10, i.e., 20 - 15 to get 5. The Mod10 check digit is 5. The Mod10 check digit for 401 is 0; for 9999, it's 4; for 99999999, it's 7.

The algorithm for calculating a Mod11 check digit is as follows:

Terms

d = digit of number starting from units digit, followed by 10's position, followed by 100's position, etc.
w = weight of digit position starting with the units position, followed by 10's position, followed by 100's position etc. Values for w = 2, 3, 4, 5, 6, 7, 2, 3, 4, 5, 6, 7, etc. (repeats for each group of 6 digits)
c =check digit

Calculation

(Step 1) m = sum of (d * w) for positions 1, 2, etc. starting with units digit

for d = digit value starting with units position to highest order
for w = weight value from 2 to 7 for every six positions starting with units digit

(Step 2)c1 = m mod 11

(Step 3)if c1 = 0 then reset c1 = 1

(Step 4)c = (11 - c1) mod 10

Example:

```
if the number is 1234567, then the mod 11 check digit = 6
```

The calculations are:

```
M  = (7*2)+(6*3)+(5*4)+(4*5)+(3*6)+(2*7)+(1*2)
   = 14 + 18 + 20 + 20 + 18 + 14 + 2
   = 106
c1 = 106 mod 11
   = 7
c  = (11-c1) mod 10
   =  4 mod 10
   =  4
```

Other variants of these check digit algorithms exist and may be used by local bilateral site agreement.

2.8.10.15 CN composite ID number and name

<ID Number> ^ <family name> ^ <given name> ^ <middle initial or name> ^ <suffix (e.g., JR or III)> ^ <prefix (e.g., DR)> ^ <degree (e.g., MD)> ^ <source table>

All components are ST data type. A field identifying a person both as a coded value and with a text name. The first component is the coded ID according to a site-specific table. The second through the seventh components are the person's name as a PN field. The eighth component specifies the source table used for the first component. For specific fields, individual sites may elect to omit the ID or the name. Example:

```
|12372^RIGGINS^JOHN^""^""^""^MD^ADT1|
|12372|
|^RIGGINS^JOHN^""^""^""^MD|
```

2.8.10.16 CQ composite quantity with units

<quantity> ^ <units>

The first component is a quantity and the second is the units in which the quantity is expressed. Field by field, default units may be defined within the specifications. When the observation is measured in the default units, the units need not be transmitted. If the measure is recorded in units different from the default, the measurement units must be transmitted as the second component. If the units are ISO+ units, then units should be recorded as lowercase abbreviations as specified in Chapter 7. If the units are ANSI or local, the units and the source table must be recorded as specified in Chapter 7. But in these cases the component separator should be replaced by the subcomponent delimiter. Examples:

```
|123.7^kg|  kilograms is an ISO unit
|150^lb&&ANS+| weight in pounds is a customary US unit defined within ANSI+.
```

In future versions, CQ fields will be used sparingly because the same data can usually be sent as two separate fields, one with the value and one with the units as a CE data type.

2.8.10.17 CE coded element

This data type transmits codes and the text associated with the code. This type has six components arranged in two groups as follows:

<identifier> ^ <text> ^ <name of coding system> ^ <alternate identifier> ^ <alternate text> ^ <name of alternate coding system>

To allow all six components of a CE data type to be valued, the minimum length of this data type is 60.

These are defined as follows:

2.8.10.17.1 **identifier:** Sequence of characters (the code) that uniquely identifies the item being referenced by the <text>. Different coding schemes will have different elements here.

2.8.10.17.2 **text:** Name or description of the item in question. E.g., myocardial infarction or x-ray impression. Its data type is string (ST).

2.8.10.17.3 **name of coding system:** Each coding system will be assigned a unique identifier. This component will serve to identify the coding scheme being used in the identifier component. The combination of the **identifier** and **name of coding system** components will be a unique code for a data item. For backward compatibility, if this component is absent, it will be taken to mean the CPT-4 with ASTM extensions, i.e., AS4. Other coding systems that might appear here are ICD-9, ICD-10, SNOMED, etc. Each system will be given a unique identifying string. The current ASTM 1238-88 diagnostic/procedure/observation/drug ID/health outcomes coding systems are identified in the tables below. Others may be added as needed.

2.8.10.17.4 **alternate components:** these three components are defined analogously to the above for the alternate or local coding system. If the Alternate Text component is absent, and the Alternate Identifier is present, the Alternate Text will be taken to be the same as the Text component. If the Alternate Coding System component is absent, it will be taken to mean the locally defined system.

Note: The presence of two sets of equivalent codes in this data type is semantically different from a repetition of a CE-type field. With repetition, several distinct codes (with distinct meanings) may be transmitted.

Example:

```
|54.21^Laparoscopy^I9^42112^^AS4|
```

Figures 2-2 and 2-3 list many diagnostic, procedure, observation, drug, and health outcomes coding systems. Guidelines for their use are presented in Section 7.1.

Figure 2-2 Diagnostic coding schemes
(from ASTM 1238-94 Table 3)

Name	Code	Source
American College of Radiology finding codes	ACR	Index for Radiological Diagnosis Revised, 3rd Edition 1986, American College of Radiology, Reston, VA.
CEN ECG diagnostic codes	CE	CEN PT007. A quite comprehensive set of ECG diagnostic codes (abbreviations) and descriptions published as a pre-standard by CEN TC251. Available from CEN TC251 secretariat, c/o Georges DeMoor, State University Hospital Gent, De Pintelaan 185-5K3, 9000 Gent, Belgium or Jos Willems, University of Gathuisberg, 49 Herestraat, 3000 Leuven, Belgium.
CLIP	CLP	Simon Leeming, Beth Israel Hospital, Boston MA. Codes for radiology reports.
EUCLIDES	E	Available from Euclides Foundation International nv, Excelsiorlaan 4A, B-1930 Zaventem, Belgium; Phone: 32 2 720 90 60.
Home Health Care	HHC	Home Health Care Classification System; Virginia Saba, EdD, RN; Geogetown University School of Nursing; Washington, DC.
ICD9	I9	World Health Publications, Albany, NY.
ICD9-CM	I9C	Commission on Professional and Hospital Activities, 1968 Green Rd., Ann Arbor, MI 48105.
ICD-10	I10	World Health Publications, Albany, NY.
Local general code	99zzz or L	Locally defined codes for purpose of sender or receiver. Local codes can be identified by L (for backward compatibility) or 99zzz (where z is an alphanumeric character).
Local billing code	LB	Local billing codes/names (with extensions if needed).
Omaha	OHA	Omaha Visiting Nurse Association, Omaha, NB.
NANDA	NDA	North American Nursing Diagnosis Association, Philadelphia, PA.
Read Classification	RC	The Read Clinical Classification of Medicine, Park View Surgery, 26 Leicester Rd., Loughborough LE11 2AG (includes drug procedure and other codes, as well as diagnostic codes).
Systemized Nomenclature of Medicine (SNOMED)	SNM	Systemized Nomenclature of Medicine, 2nd Edition 1984 Vols 1, 2, American College of Pathology, Skokie, IL.
SNOMED International	SNM3	SNOMED International, 1993 Vols 1-4, American College of Pathology, Skokie, IL.
Unified Medical Language	UML	National Library of Medicine, 8600 Rockville Pike, Bethesda, MD 20894.

Figure 2-3 Procedure/observation/drug ID/health outcomes coding systems
(From ASTM 1238-88 Table 5)

Coding System	Code	Source/Description
ASTM E1238/ E1467 Universal	AS4	American Society for Testing & Materials and CPT4 (see Appendix X1 of Specification E1238 and Appendix X2 of Specification E1467).
CPT-4	C4	American Medical Association, P.O. Box 10946, Chicago IL 60610.
CPT-5	C5	(under development - same contact as above)
EUCLIDES	E	AFP codes. Available from Euclides Foundation International nv, Excelsiorlaan 4A, B-1930 Zaventem, Belgium; Phone: 32 2 720 90 60.
FDA K10	FDK	Dept. of Health & Human Services, Food & Drug Administration, Rockville, MD 20857. (device & analyte process codes).
Health Outcomes	HI	Health Outcomes Institute codes for outcome variables available (with responses) from Health Outcomes Institute, 2001 Killebrew Drive, Suite 122, Bloomington, MN 55425; (612) 858 9188. See examples in Appendix A.
HIBCC	HB	Health Industry Business Communications Council, 5110 N. 40th St., Ste 120, Phoenix, AZ 85018.
Home Health Care	HHC	Home Health Care Classification System; Virginia Saba, EdD, RN; Geogetown University School of Nursing; Washington, DC.
ICCS	ICS	Commission on Professional and Hospital Activities, 1968 Green Road, Ann Arbor, MI 48105.
ICD-9CM	I9C	Commission on Professional and Hospital Activities, 1968 Green Road, Ann Arbor, MI 48105 (includes all procedures and diagnostic tests).
ICHPPC-2	IC2	International Classification of Health Problems in Primary Care, Classification Committee of World Organization of National Colleges, Academies and Academic Associations of General Practitioners (WONCA), 3rd edition. An adaptation of ICD9 intended for use in General Medicine, Oxford University Press.
ISBT	IBT	International Society of Blood Transfusion. Blood Group Terminology 1990. VOX Sanquines 1990 58(2):152-169.
IUPAC/IFCC	IUC	Recommendations of Quantities and Units in Clinical Chemistry DRAFT (to be published in 1992). Henrik Olesen, M.D., D.M.Sc., Chairperson, Department of Clinical Chemistry, KK76.4.2, Rigshospitalet, University Hospital of Copenhagen, DK-2200, Copenhagen.
Japanese Chemistry	JC8	Clinical examination classification code. Japan Association of Clinical Pathology. Version 8, 1990. A multiaxial code including a subject code (e.g., Rubella = 5f395, identification code (e.g., virus ab IGG), a specimen code (e.g., serum =023) and a method code (e.g., ELISA = 022)
Local	99zzz or L	Locally defined codes for purpose of sender or receiver. If multiple local codes exist, the format should be 99zzz, where z is an alphanumeric character.
Medicare	MCR	Medicare billing codes/names.
Medicaid	MCD	Medicaid billing codes/names.
Nursing Interventions Classification	NIC	Iowa Intervention Project, College of Nursing, University of Iowa, Iowa City, Iowa.

Coding System	Code	Source/Description
Omaha System	OHA	Omaha Visiting Nurse Association, Omaha, NB.
UCDS	UC	Uniform Clinical Data Systems. Ms. Michael McMullan, Office of Peer Review Health Care Finance Administration, The Meadows East Bldg., 6325 Security Blvd., Baltimore, MD 21207; (301) 966 6851.
Universal Product Code	UPC	The Uniform Code Council. 8163 Old Yankee Road, Suite J, Dayton, OH 45458; (513) 435 3070
Euclides Lab method codes	E6	Available from Euclides Foundation International nv, Excelsiorlaan 4A, B-1930 Zaventem, Belgium; Phone: 32 2 720 90 60.
Euclides Lab equipment codes	E7	Available from Euclides Foundation International nv (see above)
Euclides quantity codes	E5	Available from Euclides Foundation International nv (see above)
Drug codes:		
Chemical abstract codes	CAS	These include unique codes for each unique chemical, including all generic drugs. The codes do not distinguish among different dosing forms. When multiple equivalent CAS numbers exist, use the first one listed in USAN. USAN 1990 and the USP dictionary of drug names, William M. Heller, Ph.D., Executive Editor, United States Pharmacopeial Convention, Inc., 12601 Twinbrook Parkway, Rockville, MD 20852.
National drug codes	NDC	These provide unique codes for each distinct drug, dosing form, manufacturer, and packaging. (Available from the National Drug Code Directory, FDA, Rockville, MD, and other sources.)
WHO rec# drug codes	W1, W2	World Health organization record number code. A unique sequential number is assigned to each unique single component drug and to each multi-component drug. Eight digits are allotted to each such code, six to identify the active agent, and 2 to identify the salt, of single content drugs. Six digits are assigned to each unique combination of drugs in a dispensing unit. The six digit code is identified by W1, the 8 digit code by W2.
WHO rec# code with ASTM extension	W4	With ASTM extensions (see appendix X1), the WHO codes can be used to report serum (and other) levels, patient compliance with drug usage instructions, average daily doses and more (see Appendix X1). WHO ATC code WC WHO's ATC codes provide a hierarchial classification of drugs by therapeutic class. They are linked to the record number codes listed above.
WHO ATC	WC	WHO's ATC codes provide a hierarchical classification of drugs by therapeutic class. They are linked to the record number codes listed above.
Note: The Read and NLM (National Library of Medicine) codes in Table 3 also include drugs. A number of sources of unique drug names exist: British Approved Names (BAN), French-approved nonproprietary names (DCF), and International Nonproprietary name (INN). These sources are now being reviewed. Those that also provide unique codes will be added to the registry of coding systems, using the abbreviations given in parentheses.		
Device Code		

Coding System	Code	Source/Description
MDNS	UMD	Universal Medical Device Nomenclature System. ECRI, 5200 Butler Pike, Plymouth Meeting, PA 19462 USA. Phone: 215-825-6000, Fax: 215-834-1275.

2.8.10.18 CF coded element with formatted values

This data type transmits codes and the formatted text associated with the code. This data type can be used to transmit for the first time the formatted text for the **canned text** portion of a report, for example, a standard radiologic description for a normal chest x-ray. The receiving system can store this information and in subsequent messages only the identifier need be sent. Another potential use of this data type is transmitting master file records that contain formatted text.

This data type has six components arranged in two groups as follows:

```
<identifier> ^ <formatted text> ^ <name of coding system>^
<alternate identifier> ^ <alternate formatted text> ^ <name of alternate coding system>
```

The components, primary and alternate are defined exactly as in the CE data type with the exception of the second and fifth components which are of the formatted text data type. Example:

```
OBX||CF|71020^CXR^CPMC||79989^
\H\Description:\N\\.sp\\ti+4\Heart is not enlarged.  There is no evidence of
pneumonia, effusion, pneumothorax or any masses.\.sp+3\\H\Impression:
\N\\.sp\\.ti+4\Negative chest.^CPMC
```

2.8.10.19 RP reference pointer

This data type transmits information about data stored on another system. It contains a reference pointer that uniquely identifies the data on the other system, the identity of the other system, and the type of data. This information is transmitted in three components arranged as follows:

```
<pointer> ^ <application ID> ^ <type of data>
```

2.8.10.19.1**pointer:** A unique key assigned by the system that stores the data. The key, which is an ST data type, is used to identify and access the data.

2.8.10.19.2**application ID:** A unique name (up to 6 characters in length) of the system that stores the data. It is an ST data type. It is equivalent to the application ID of the placer or filler order number (see Chapter 4). Application ID's must be unique across a given HL7 implementation.

2.8.10.19.3**type of data:** A code that represents the type of data being stored. It is an ID data type.

Table 0191 Type of Data

Value	Description
SI	Scanned image
NS	Non-scanned image
SD	Scanned document
TX	Machine readable text document
FT	Formatted text

Other types may be added as needed.

Example:

```
|1234A321634BC^EFC^SD|
```

2.8.10.20 TQ timing quantity

Describes when a service should be performed and how frequently. See Chapter 4 (Section 4.4) for a complete specification.

2.8.10.21 MO money

```
<quantity> ^ <denomenation>
```

The first component is a quantity and the second is the denomination in which the quantity is expressed. The values for the denomination component are those specified in ISO-4217. If the denomination is not specified, *MSH-17-country code* is used to determine the default.

Example:

```
|99.50^USD|
```

where USD is the ISO 4217 code for the U.S. American dollar.

2.8.11 Use of escape sequences in text fields

2.8.12.1 Formatting codes

When a field of type TX, FT, or CF is being encoded, the escape character may be used to signal certain special characteristics of portions of the text field. The escape character is whatever display ASCII character is specified in the Escape Character component of *MSH-2-encoding characters*. For purposes of this section, the character \ will be used to represent the character so designated in a message. An **escape sequence** consists of the escape character followed by an escape code ID of one character, 0 or more data characters, and another occurrence of the escape character. The following escape sequences are defined:

\H\	start highlighting
\N\	normal text (end highlighting)
\F\	field separator
\S\	component separator

\T\	subcomponent separator
\R\	repetition separator
\E\	escape character
\Xdddd\	hexadecimal data
\Zdddd\	locally defined escape sequence

The **escape sequences** for field separator, component separator, subcomponent separator, repetition separator, and escape character are also valid within an ST data field.

No escape sequence may contain a nested escape sequence.

2.8.12.2 Highlighting

In designating highlighting, the sending application is indicating that the characters that follow somehow should be made to stand out, but leaving the method of doing so to the receiving application. Depending on device characteristics and application style considerations, the receiving application may choose reverse video, boldface, underlining, blink, an alternate color or another means of highlighting the displayed data. For example the message fragment:

```
DSP|    TOTAL CHOLESTEROL    \H\240*\N\   [90 - 200]
```

might cause the following data to appear on a screen or report:

```
TOTAL CHOLESTEROL       240*    [90 - 200]
```

whereas another system may choose to show the 240* in red.

2.8.12.3 Special character

The special character escape sequences (\F\, \S\, \R\, \T\, and \E\) allow the corresponding characters to be included in the data in a text field, though the actual characters are reserved. For example, the message fragment

```
DSP| TOTAL CHOLESTEROL     180  \F\90 - 200\F\
DSP| \S\----------------\S\
```

would cause the following information to be displayed, given suitable assignment of separators:

```
TOTAL CHOLESTEROL     180  |90 - 200|
^----------------^
```

2.8.12.4 Hexadecimal

When the hexadecimal escape sequence (\Xdddd\) is used the X should be followed by 1 or more pairs of hexadecimal digits (0, 1, . . . , 9, A, . . . , F). Consecutive pairs of the hexadecimal digits represent 8-bit binary values. The interpretation of the data is entirely left to an agreement between the sending and receiving applications that is beyond the scope of this Standard.

2.8.12.5 Formatted text

If the field is of the formatted text (FT) data type, formatting commands also may be surrounded by the escape character. Each command begins with the . character. The following formatting commands are available:

.sp <number>End current output line and skip <number> vertical spaces. <number> is a positive integer or absent. If <number> is absent, skip one space.

.br Begin new output line.

.fi Begin word wrap or fill mode. This is the default state. It can be changed to a no-wrap mode using the .nf command.

.nf Begin no-wrap mode.

.in <number>Indent <number> of spaces, where <number> is a positive or negative integer.

.ti <number>Temporarily indent <number> of spaces where number is a positive or negative integer.

.ce End current output line and center the next line.

The component separator that marks each line defines the extent of the temporary indent command (.ti), and the beginning of each line in the no-wrap mode (.nf). Examples of formatting instructions that are NOT included in this data type include: width of display, position on page or screen, and type of output devices.

Figure 2-4 is an example of the FT data type from a radiology impression section of a radiology report:

Figure 2-4 Formatted text as transmitted

```
|\.in+4\\.ti-4\ 1. The cardio mediastinal silhouette is now within normal limits.^\.sp\\.ti-4\ 2. Lung fields show minimal ground glass
        appearance.^\.sp\\.ti-4\ 3. A loop of colon visible in the left upper quadrant is distinctly abnormal with the appearance of
        mucosal effacement suggesting colitis.\.in-4\|
```

Figure 2-5 shows one way of presenting the data in Figure 2-4. The receiving system can create many other interpretations by varying the right margin.

Figure 2-5 Formatted text in one possible presentation

```
1.The cardio mediastinal silhouette is now within normal limits.

2.Lung fields show minimal ground glass appearance.

3.A loop of colon visible in the left upper quadrant is distinctly abnormal with the appearance of mucosal effacement
          suggesting colitis.
```

2.8.12.6 Local

When the local escape sequence (\Zdddd\) is used the Z should be followed by characters that are valid in a TX field. The interpretation of the data is entirely left to an agreement between the sending and receiving applications that is beyond the scope of this Standard.

2.8.13 Message construction rules

Step 1Construct the segments in the order defined for the message. Each message is constructed as follows:

a)the first three characters are the segment ID code

b)each data field in sequence is inserted in the segment in the following manner:

1)a field separator is placed in the segment

2)if the value is not present, no further characters are required [1]

3)if the value is present, but null, the characters "" (two consecutive double quotation marks) are placed in the field [1]

4)otherwise, place the characters of the value in the segment. As many characters can be included as the maximum defined for the data field. It is not necessary, and is undesirable, to pad fields to fixed lengths. Padding to fixed lengths is permitted.

Encode the individual data fields as shown in the Data Types section (2.4.5).

5)if the field definition calls for a field to be broken into components, the following rules are used:

i)if more than one component is included they are separated by the component separator

ii)components that are present but null are represented by the characters ""

iii)components that are not present are treated by including no characters in the component

[1]The receiving system, if storing the field in its database, distinguishes these two conditions as follows:
field value not present - do not change value of field in database
field value present but null - change database value of field to null

iv)components that are not present at the end of a field need not be represented by component separators. For example, the two data fields are equivalent:

|ABC^DEF^^| and |ABC^DEF|.

6)if the component definition calls for a component to be broken into subcomponents, the following rules are used:

i)if more than one subcomponent is included they are separated by the subcomponent separator

ii)subcomponents that are present but null are represented by the characters ""

iii)subcomponents that are not present are treated by including no characters in the subcomponent

iv)subcomponents that are not present at the end of a component need not be represented by subcomponent separators. For example, the two data components are equivalent:

^XXX&YYY&&^ and ^XXX&YYY^.

7)if the field definition permits repetition of a field, the following rules are used, the repetition separator is used only if more than one occurrence is transmitted and is placed between occurrences. (If three occurrences are transmitted, two repetition separators are used.) In the example below, two occurrences of telephone number are being sent:

|234-7120~599-1288B1234|

c)repeat Step 1b while there are any data elements present to be sent. If all the data fields remaining in the segment definition are not present there is no requirement to include any more delimiters.

d)end each segment with an ASCII carriage return character

Step 2Repeat Step 1 until all segments have been generated.

The following rules apply to receiving HL7 messages and converting their contents to data values:

a)ignore segments, fields, components, subcomponents, and extra repetitions of a field that are present but were not expected

b)treat segments that were expected but are not present as consisting entirely of fields that are not present

c)treat fields and components that are expected but were not included in a segment as not present

d)in applications that cannot deal with the distinction between data fields that are not present and those that are null, treat data fields that are not present as null

Encoding rules notes:

If a segment is to be continued across messages, use the extended encoding rules given below. These rules are defined in terms of the more general message continuation protocol (described below in Section 2.9.2-continuation messages and segments).

2.8.15 Chapter formats for defining HL7 messages

Subsequent chapters of this document describe messages that are exchanged among applications in functionally specific situations. Each chapter is organized as follows:

a)purpose. This is an overview describing the purpose of the chapter, general information and concepts.

b)trigger events and messages. There is a list of the trigger events. For each trigger event the messages that are exchanged when the trigger even occurs are defined using the **HL7 abstract message syntax** as follows:

Each message is defined in special notation that lists the segment IDs in the order they would appear in the message. Braces, **{ . . . }**, indicate one or more repetitions of the enclosed group of segments. (Of course, the group may contain only a single segment.) Brackets, **[. . .]**, show that the enclosed group of segments is optional. If a group of segments is optional and may repeat it should be enclosed in brackets and braces, **{ [. . .] }**.

Note:[{...}] and {[...]} are equivalent.

Whenever braces or brackets enclose more than one segment ID a special stylistic convention is used to help the reader understand the hierarchy of repetition. For example, the first segment ID appears on the same line as the brace, two columns to the right. The subsequent segment IDs appear under the first. The closing brace appears on a line of its own in the same column as the opening brace. This convention is an optional convenience to the user. If there is conflict between its use and the braces that appear in a message schematic, the braces define the actual grouping of segments that is permitted.

c)message segments. The segments defined in a chapter are then listed in a functional order designed to maximize conceptual clarity.

d)examples. Complete messages are included.

e)implementation considerations. Special supplementary information is presented here. This includes issues that must be addressed in planning an implementation.

f)outstanding issues. Issues still under consideration or requiring consideration are listed here.

Consider the hypothetical triggering event **a widget report is requested**. It might be served by the Widget Request (WRQ) and Widget Report (WRP) messages. These would be defined in the Widget chapter (say Chapter XX). The Widget Request message might consist of the following segments: Message Header (MSH), Widget ID (WID). The Widget Report message might consist of the following segments: Message Header (MSH), Message Acknowledgement (MSA), one or more Widget

Description (WDN) Segments each of which is followed by a single Widget Portion segment (WPN) followed by zero or more Widget Portion Detail (WPD) segments.

2.8.16.1 HL7 abstract message syntax example

The schematic form for this hypothetical exchange of messages is shown in Figure 2-6:

Figure 2-6 Hypothetical schematic message

```
Trigger Event: WIDGET REPORT IS REQUESTED

     WRQ        Widget Request         Chapter
     MSH        Message Header           2
     WID        Widget ID              XX

     WRP        Widget Report          Chapter
     MSH        Message Header           2
     MSA        Message Acknowledgement    2
     { WDN      Widget Description      XX
       WPN      Widget Portion         XX
       { [WPD] }  Widget Portion Detail   XX
     }
```

The WID, WDN, WPN, and WPD segments would be defined by the widget committee in the widget chapter, as designated by the arabic numeral XX in the right column. The MSH and MSA segments, although included in the widget messages, are defined in another chapter. They are incorporated by reference into the widget chapter by the chapter number XX.

On the other hand, the widget committee might decide that the WPN and WPD segments should appear in pairs, but the pairs are optional and can repeat. Then the schematic for the WRP message would be as shown in Figure 2-7.

Figure 2-7 WPN and WPD segments in pairs

```
     WRP        Widget Report           Chapter

     MSH        Message Header             2
     MSA        Message Acknowledgement      XX
     { WDN      Widget Description         XX
       [ { WPN  Widget Portion            XX
           WPD  Widget Portion Detail       XX
       } ]
     }
```

If the widget committee determined that at least one pair of WPN and WPD segments must follow a WDN, then the notation would be as shown in Figure 2-8.

Figure 2-8 At least one pair of WPN and WPD

```
     WRP        Widget Report           Chapter
```

```
MSH        Message Header              2
MSA        Message Acknowledgement       XX
{ WDN       Widget Description          XX
  {  WPN     Widget Portion            XX
     WPD     Widget Portion Detail       XX
  }
}
```

2.9 APPLICATION (LEVEL 7) PROCESSING RULES

2.10.1 Original and enhanced processing rules

The processing rules described here apply to all exchanges of messages, whether or not the HL7 encoding rules or Lower Layer Protocols are used. They represent the primary message processing mode. Certain variants are documented elsewhere in the next section. These include:

a)the application processing rules for a special processing mode, deferred processing. This mode remains in the specification only for backwards compatibility.

b)an optional sequence number protocol

c)an optional protocol for continuing a very long message

The processing rules have been extended in this version of the standard. The extensions provide a greater degree of flexibility in the way that messages can be acknowledged, as specified by several new fields in the Message Header segment. To provide backwards-compatibility with prior versions, the absence of these fields implies that the extended processing rules are not used. In the remainder of this section the extended mode is called the enhanced acknowledgement mode; the prior version is called the original acknowledgement mode.

Because the protocol describes an exchange of messages, it is described in terms of two entities, the **initiating** and **responding** systems. Each is both a sender and receiver of messages. The initiating system sends first and then receives, while the responding system receives and then sends.

In overview this exchange proceeds as follows:

Step 1the initiating system constructs an HL7 message from application data and sends it to the responding system

Step 2responder receives message and

2.1when the original acknowledgement rules apply:

a)validates it syntactically and against the detailed rules described in Section 2.5.1.1, if it fails, a reject message is constructed by the protocol software and returned to the initiator

b) passes it to the application, which:

1)creates a response message, or

2)creates an error message, or

3)creates a reject message

c) sends the response, error, or reject message

Initiator passes the message to the initiating application.

2.2when enhanced acknowledgement rules apply:

a)the responding system receives the message and commits it to safe storage. This means that the responding system accepts the responsibility for the message in a manner that releases the sending system from any obligation to resend the message. The responding system now checks the message header record to determine whether or not the initiating system requires an **accept acknowledgement** message indicating successful receipt and secure storage of the message. If it does, the accept acknowledgement message is constructed and returned to the initiator.

b)at this point, the requirements of the applications involved in the interface determine whether or not more information needs to be exchanged. This exchange is referred to as an **application acknowledgement** and includes information ranging from simple validation to a complex application-dependent response. If the receiving system is expected to return application-dependent information, it initiates another exchange when this information is available. This time, the roles of initiator and responder are reversed.

The details follow.

2.10.2.1 Initiation

The initiating application creates a message with data values as defined in the appropriate chapter of this Standard.

The data values shown below should be provided in the MSH segment (as defined under the MSH segment definition of this chapter). The message is encoded according to the applicable rules and sent to the lower level protocols, which will attempt to deliver it to the responding application.

Field	**Contents**
MSH-7-date/time of message	The date and time the data values are combined into the message. This field is not used in the processing logic of the HL7 protocol. It is optional.
MSH-9-message type	A two-component field. The first component is a three-letter code identifying this message (as defined above). The second component is a three-digit trigger event code as listed in *table 0003 - event type code*.
MSH-5-receiving application	A code defined for the communication environment to identify the application to which the message should be sent.
MSH-3-sending application	A code similar to Receiving Application used to identify the source of the message.
MSH-6-receiving facility	A code defined for the communication environment to identify which instance of the application should receive the message.
MSH-4-sending facility	A code similar to Receiving Facility used to identify the source of the message.
MSH-10-message control ID	Unique identifier used to relate the response to the initial message.
MSH-11-processing ID	A code that show whether this transaction is to be used for production, training, debugging.
MSH-12-version ID	The version number of the HL7 spec for which the software was defined.
MSH-13-sequence number	Sequence number: used in implementation of sequence number protocol. See Section 2.9.1-sequence number protocol.
MSH-14-continuation pointer	Used in implementation of message continuation protocol. See sections 2.9.2-continuation messages and segments, 2.7.3-continuation of unsolicited display update message, and 2.8.4-interactive continuation of display messages.

Certain other fields in the MSH segment are required for the operation of the HL7 encoding rules; they will not be relevant if other encoding rules are employed.

The event code in the second component of *MSH-9-message type* is redundantly shown elsewhere in some messages. For example, the same information is in the EVN segment of the ADT message. This is for compatibility with prior versions of the HL7 protocol. Newly defined messages should only show the event code in *MSH-9-message type*.

2.10.2.2 Response

The protocol software in the responding system does one of the following:

2.10.2.2.1 When the original acknowledgement rules apply

> **Note:**Both *MSH-15-accept acknowledgement type* and *MSH-16-application acknowledgement type* are null or not present.

a)accepts the message

b)validates it against at least the following criteria:

1)the value in *MSH-9-message type* is one that is acceptable to the receiver

2)the value in *MSH-12-version ID* is acceptable to the receiver

3)the value in *MSH-11-processing ID* is appropriate for the application process handling the message

If any of these edits fail, the protocol software rejects the message. That is, it creates an ACK message with **AR** in *MSA-1-acknowledgement code.*

c)if the message passes the edits, the message is passed to the receiving application, which performs one of these functions:

1)process the message successfully, generating the functional response message with a value of **AA** in *MSA-1-acknowledgement code*.

- OR -

2)send an error response, providing error information in functional segments to be included in the response message with a value of **AE** in *MSA-1-acknowledgement code*.

- OR -

3)fail to process (reject) the message for reasons unrelated to its content or format (system down, internal error, etc.). For most such problems it is likely that the responding system will be able to accept the same message at a later time. The implementors must decide on an application-specific basis whether the message should be automatically sent again. The response message contains a value of **AR** in *MSA-1-acknowledgement code.*

d) passes the message to the initiating system

e)the protocol software in the initiating system passes the response message to the initiating application

In all the responses described above the following values are put in the MSA segment:

Field	**Contents**
MSA-2-message control ID	*MSH-10-message control ID* from MSH segment of incoming message.
MSA-1-acknowledgement code	As described above.
MSA-3-text message	Text description of error.
MSA-4-expected sequence number	As described in Section 2.9.1 - Sequence Number Protocol, (if the sequence number protocol is being used).
MSA-5-delayed acknowledgement type	For use only as described in Section 2.5.2-application (level 7) processing rules, deferred processing two phase reply.

The MSH segment in the response is constructed anew following the rules used to create the initial message described above. In particular, *MSH-7-date/time of message* and *MSH-10-message control ID* refer to the response message; they are not echoes of the fields in the initial message. *MSH-5-receiving application*, *MSH-6-receiving facility*, and *MSH-11-processing ID* contain codes that are copied from *MSH-3-sending application*, *MSH-4-sending facility* and *MSH-11-processing ID* in the initiating message.

2.10.2.2.2 When enhanced acknowledgement rules apply

Note: At least one of *MSH-15-accept acknowledgement type* or *MSH-16-application acknowledgement type* is not null.

a)accepts the message

b)makes an initial determination as to whether or not the message can be accepted, based on factors such as:

1)the status of the interface

2)the availability of safe storage onto which the message can be saved

3)the syntactical correctness of the message, if the design of the receiving system includes this type of validation at this phase

4)the values of *MSH-9-message type*, *MSH-12-version ID*, and *MSH-11-processing ID*, if the design of the receiving system includes this type of validation at this phase

c)examines the Message Header segment (MSH) to determine whether or not the initiating system requires an accept acknowledgement.

If it does, the responding system returns a general acknowledgement message (ACK) with:

1)a commit accept (CA) in *MSA-1-acknowledgement code* if the message can be accepted for processing

2)a commit reject (CR) in *MSA-1-acknowledgement code* if the one of the values of *MSH-9-message type*, *MSH-12-version ID* or *MSH-11-processing ID* is not acceptable to the receiving application

3)a commit error (CE) in *MSA-1-acknowledgement code* if the message cannot be accepted for any other reason (e.g., sequence number error)

For this response, the following values are put in the MSA segment:

Field	**Contents**
MSA-2-message control ID	*MSH-10-message control ID* from the incoming message.
MSA-1-acknowledgement code	As described above.
MSA-3-text message	Text description of error.
MSA-4-expected sequence number	As described in Sequence Number Protocol, below, (if the sequence number protocol is being used).

The MSH segment in the response is constructed anew following the rules used to create the initial message described above. In particular, *MSH-7-date/time of message* and *MSH-10-message control ID* refer to the response message; they are not echoes of the fields in the initial message. *MSH-5-receiving application*, *MSH-6-receiving facility*, and *MSH-11-processing ID* contain codes that are copied from *MSH-3-sending application*, *MSH-4-sending facility* and *MSH-11-processing ID* in the initiating message.

Note: *MSH-15-accept acknowledgement type* and *MSH-16-application acknowledgement type* are not valued (not present or null). At this point, the accept portion of this message exchange is considered complete.

d)If the message header segment indicates that the initiating system also requires an application acknowledgement, this will be returned as the initial message of a later exchange.

For this response, the following values are put in the MSA segment:

Field	**Contents**
MSA-2-message control ID	Identifies the initial message from the original initiating system as defined in Section 2.5.1.1-initiation.
MSA-1-acknowledgement code	Uses the application (processing) acknowledgement codes as described in Section 2.5.1.2.1-when the original acknowledgement rules apply.
MSA-3-text message	Text description of error.

For this message, the receiving system acts as the initiator. Since the message it sends is application-specific, the layouts of these application-level response messages are defined in the relevant application-specific chapter. If needed, this application acknowledgement message can itself require (in *MSH-15-accept acknowledgement type*) an accept acknowledgement message (MSA). *MSH-16-application acknowledgement type*, however, is always null, since the protocol does not allow the application acknowledgement message to have an application acknowledgement.

At this point, the application acknowledgement portion of this message exchange is considered complete.

If the processing on the receiving system goes through multiple stages, chapter-defined messages may be used to relay status or informational changes to other systems (including the original initiating system). Such messages are not part of the acknowledgement scheme for the original message, but are considered to be independent messages triggered by events on the (original) responding system.

Note: The original acknowledgement protocol is equivalent to the enhanced acknowledgement protocol with *MSH-15-accept acknowledgement type* = NE and *MSH-16-application acknowledgement type* = AL, and with the application acknowledgement message defined so that it never requires an accept acknowledgement (*MSH-15-accept acknowledgement type* = NE).

2.10.3Application (level 7) processing rules, deferred processing two phase reply (original acknowledgement mode only)

(This section remains in the specification only for reasons of providing backwards compatibility: it is to be used only with the original acknowledgement protocol. For the original acknowledgement protocol, it creates a generic form of an asynchronous application level acknowledgement, the MCF message).

The application processing rules for deferred processing are described here. In this mode the responding system sends an acknowledgement to the initiating system that means the message has been placed in some type of secure environment (e.g., disk storage), and the receiving system commits to processing it within a reasonable amount of time, if a) the message contains the necessary information, and b) nothing causes the message's request for action to be canceled before the responding system processes the request.

Note: Neither of these two conditions is completely checked at the time of the first acknowledgement. They are both checked at the time of processing.

The receipt of the first delayed acknowledgement by the initiating system means that the responding system has taken responsibility for the subsequent processing of the message. This also implies that the initiating system no longer needs to keep the particular message in its current form to send out later. For example, if the sending system were maintaining a queue of outgoing messages, the particular message could be deleted from the output queue at this point.

The receipt of the second delayed acknowledgement message informs the initiating application of either a) the application's successful processing of the initial message, or b) an error that prevented its processing. If the receiving application needs to return detailed change of status information, an application-specific message will be used. An example of the latter is the General Order message (ORM) described in Chapter 4.

The general delayed acknowledgement protocol is implemented on a site-specific and application-specific basis as needed. At a particular site, for a given transaction type the choices are:

a)do not allow deferred acknowledgements

b)all messages will have a deferred acknowledgement

c)only exceptional cases (errors) will receive the deferred acknowledgement

In overview the processing for options b) and c) proceeds as follows:

Initiator receives message from sending application and sends it to the responding system.

The responding system receives the message from the initiating system and

a)partially validates it syntactically and against the detailed rules described in Section 2.5.1, under original acknowledgement mode. This validation need not be complete but should be sufficient to determine the application that will ultimately respond to the message. If this validation fails, a reject message is constructed by the protocol software and returned to the initiator.

b)(if the message passes this validation) stores it and constructs a response message that simply acknowledges receipt. *MSA-5-delayed acknowledgement type* then has a value of **D**.

c)subsequently passes the message to the application, which:

1)creates a response message, or

2)creates an error message, or

3)creates a reject message

d)The protocol software sends the response, error, or reject message to the initiating system as an unsolicited update with no value present in *MSA-5-delayed acknowledgement type*.

The protocol software of the initiating system responds to the response, error, or reject message with simple acknowledgement and passes it to the initiating application.

The details follow.

2.10.4.1 Initiation

The rules for creating the initial message are exactly as defined in Section 2.5.1 for the original acknowledgement rules.

2.10.4.2 Response

The processing in the responding system follows this pattern:

a)the protocol software accepts the message and validates it against at least the following criteria:

1)the value in *MSH-9-message type* is one that is acceptable to the receiver

2)the value in *MSH-12-version ID* is acceptable to the receiver

3)the value in *MSH-11-processing ID* is appropriate for the application process handling the message

If any of these edits fail, the protocol software rejects the message. That is, it creates an ACK message with **AR** in *MSA-1-acknowledgement code*.

b)If the message passes the edits, the protocol software stores it and a generates a response message of type ACK with a value of **AA** in *MSA-1-acknowledgement code* and **D** in *MSA-5-delayed acknowledgement type*.

c)Subsequently the protocol software passes the message to the application, which performs one of these functions:

1)processes the message successfully, generating the functional response message (message type MCF) with a value of **AA** in *MSA-1-acknowledgement code*.

- OR -

2)creates an error response, providing error information in functional segments to be included in the response message, which has a value of **AE** in *MSA-1-acknowledgement code*.

- OR -

3)fails to process (rejects) the message for reasons unrelated to its content or format (system down, internal error, etc.) For most such problems it is likely that the responding system will be able to accept the same message at a later time. The implementors must decide on an application-specific basis whether the message should be automatically sent again. The MSA segment of the response message contains a value of **AR** in *MSA-1-acknowledgement code*.

d)the application passes the message to the protocol software, which constructs a message of type MCF with **F** in *MSA-5-delayed acknowledgement type*.

e)the protocol software passes the message to the initiating system as an unsolicited update

f)the protocol software in the initiating system passes the response message to the initiating application and generates a simple ACK message. No value is present in *MSA-5-delayed acknowledgement type*.

All other values are put in the MSA segment as described in Section 2.5.1 under the original processing rules.

2.11 ACKNOWLEDGEMENT MESSAGES

Acknowledgement messages may be defined on an application basis. However the simple general acknowledgement message (ACK) may be used where the application does not define a special message (application level acknowledgement) and in other cases as described in Section 2.5.1 - original and enhanced processing rules. The MCF message is included only for backwards compatibility.

2.12.1 ACK: general acknowledgement

The simple general acknowledgement (ACK) can be used where the application does not define a special application level acknowledgement message or where there has been an error that precludes application processing. It is also used for accept level acknowlegements. The details are described in Section 2.5.1 - original and enhanced processing rules.

ACK	General Acknowledgement	Chapter
MSH	Message Header	2

MSA	Message Acknowledgement	2
[ERR]	Error	2

2.12.3 MCF: Delayed acknowledgement

(This message remains in the specification only for reasons of backwards compatibility. It is used as part of the protocol which creates a generic form of an asynchronous application level acknowledgement, the MCF message. See Section 2.5.2)

The first MCF message, sent after the initial receipt has the following structure.

MCF	General Acknowledgement	Chapter
MSH	Message Header	2
MSA	Message Acknowledgement	2
[ERR]	Error	2

The second MCF message, sent after application processing, has this structure:

MCF	General Acknowledgement	Chapter
MSH	Message Header	2
MSA	Message Acknowledgement	2
[ERR]	Error	2

2.13 DISPLAY MESSAGES

2.14.1 Display vs. record-oriented messages

HL7 messages may contain:

a)data in a format suitable for display purposes (display-oriented data), or

b)data in a format which explicitly denotes field content (record oriented)

A display message can be generated to fit a variety of needs for unsolicited updates between systems. These are situations where the update information does not need to be captured by the receiving system's database, but only displayed, either on a visual medium (such as a PC, workstation or a CRT) or on printed medium.

The Unsolicited Display Message describes the display oriented message. It is the unsolicited version of the generalized Response display message (see section on Queries, below). It is acknowledged by a general acknowledgement message (ACK).

The content and format of record oriented messages require functionally specific capabilities. The technical committees responsible for functionally specific chapters define them within those chapters.

2.14.3 Unsolicited display update message (trigger event Q05)

There is a simple HL7 message that allows for unsolicited display update messages to be sent in HL7 format from one system to another.

Trigger events for the unsolicited update are generally the completion of a particular action (concerning a given patient). For example, a lab test might be completed, generating a STAT unsolicited display message to be sent to the appropriate location.

UDM	Unsolicited Display Message	Chapter
MSH	Message Header	2
URD	Results/update definition	2
[URS]	Results/update selection criteria	2
{ DSP }	Display data	2
[DSC]	Continuation Pointer	2

ACK	General Acknowledgement	Chapter
MSH	Message Header	2
MSA	Message Acknowledgement	2
[ERR]	Error	2

2.14.5 Continuation of unsolicited display update message

Like other types of HL7 messages, the UDM message can be continued by use of the DSC segment and *MSH-14-continuation pointer*. Thus if a UDM needs to be continued as three separate UDM messages, the first message would contain:

```
MSH         (no continuation pointer)
URD
[URS]
{ DSP }
DSC         (with continuation pointer)
```

The second message would contain:

```
MSH         (continuation pointer (to first message))
{ DSP }
DSC         (with continuation pointer)
```

The last message would then contain:

```
MSH         (continuation pointer (to second message))
{ DSP }
            (no DSC, since last)
```

Note: This scheme works equally well with non-display messages, such as the Unsolicited Update ORU message (see

	Chapter 7).

Since these are unsolicited messages, intervening messages (from other systems) may be sent to the receiving application while the sections of the particular message are being continued. *MSH-14-continuation pointer* enables the receiving system to keep track of extraneous intervening messages.

2.15 QUERIES

2.16.1 Display vs. record-oriented queries

Responses to queries exist in both display and record-oriented format (see Section 2.7.1). This section will discuss only display-oriented responses to queries. The content and format of record oriented responses to queries require functionally specific capabilities. The technical committees responsible for functionally specific chapters define them within those chapters.

2.16.3 Message definition

This section defines two messages. The first is a generalized Query Message, intended to support most types of queries between two systems. The second is a generalized Response Display Message in which the responding system formats the data for direct output to a display.

The following represents typical examples of queries supported by the Standard:

a)for data regarding a single patient, e.g., send all lab results for patient #123456

b)for data regarding multiple patients, e.g., send the list of patients whose attending physician is Dr. #123

c)for data that is not patient related, e.g., send the age specific normal values for serum protein

The variety of potential queries is almost unlimited. There was no attempt here to define a Standard that would cover every possible query. Instead, the Standard embraces the most common queries that are likely to occur in a hospital. For each common query there is a corresponding unsolicited update. (See Section 2.7.2 above).

In particular, there is no implication that a specific system must support generalized queries to comply with the Standard. Rather, these transactions provide a format, or a set of tools to support queries to the extent desired by the institution. The resources available and local policies will influence the type of queries that are implemented.

2.16.5 Immediate vs. deferred response

Responses to queries can be either immediate or deferred. The query describes this as the expected response time. In the immediate mode, the responding process gives the response immediately or in a short period during which the requesting process will wait for the response.

2.16.7 Interactive continuation of display messages

One use of queries is to retrieve data from one application for presentation to users of another. This approach might be used for users of a patient care system retrieving data from lab or other ancillaries. It also might permit users of a pharmacy system to retrieve a patient's lab results from the lab system or non-pharmacy order data from the patient care system. Almost any other application system could be the source of data or the system initiating the query for its users.

Of particular interest is the case where the enquiring user formulates the query on-line at the terminal of one system and waits while that system sends the query to another. He gets the response and displays it at their terminal. When the user is formulating such a query she may only have limited understanding of what data is available for a given patient. Sometimes the user's preference would be to make a simple query such as **give me recent data in reverse chronological sequence** rather than **give me data for yesterday**, since there may be some data available for today, or there may be data from two days ago that is of interest. The user will look at the data returned and simply quit looking at it after he or she has found the part that is of interest. (The time frames or the sort sequence may differ, or the user may wish to impose some selectivity on the response, but the general principle remains the same. The user would prefer to make a vague statement of the interest, have the data presented in order of decreasing likelihood of interest, and quit when he or she has seen enough.)

While beneficial to the user, this way of requesting data could be very burdensome when the resulting query takes place over an inter-application interface. If the responding system were to retrieve, format, and send all the data the user might like to see, the processing load would be extremely high and the response time unacceptable.

The continuation query provides a way to permit the users to formulate queries loosely while limiting the processing burden on the responding system. The initial query specifies the general constraints of the query and an amount of data to be returned. (For example, the query might be for lab results for patient #12379 and 44 lines would be requested.) The responding system retrieves and formats the specified amount of data and returns it with a special key field, *DSC-1-continuation pointer*. The initiating system presents the requested data to the user and retains the continuation pointer field for use if another query is needed. The internal structure of the value is not known to the initiating system.

If, after viewing the data, the user requests more, the initiating system sends the query again in a format that is identical with the first, except that the *DSC-1-continuation pointer* value is included and the requested amount of data may be changed. The responding system uses the continuation pointer field as a key into its database to continue retrieval and formatting of the results. If the user does not request more data, no further messages are exchanged.

2.16.9 Logical display break points

Often the lines of display text will fall into logical groups that differ from the physical size of screen or printer page. For example, a complete battery or an entire radiology report might be thought of as comprising a logical group, though they might have as few as six or as many as 120 lines. Knowledge of the logical break points in the display data can be useful to the application system that is displaying or printing data. For this reason, *DSP-4-logical break point* is used. The sending application (the one that formats the data) places the logical break points where appropriate. If there is a particular ancillary result ID associated with the data delineated by *DSP-4-logical break point*, the value of this ID also can be returned in *DSP-5-result ID*. Then if the user selects the area of the display delineated by *DSP-4-logical break point*, the displaying system can query for the associated *DSP-5-result ID*.

2.16.11 Query trigger events and message definitions

These are the trigger events associated with queries:

a)a need occurs for immediate access to data that may be available from another application, this may be an initial request for data or a continuation

b)a need occurs for deferred access to data that may be available from another application

When display data is involved, these trigger events are served by the Query (QRY) and Display Response (DSR) and General Acknowledgement (ACK) messages. When the query is for record-oriented data, the QRY message is used, but the response message is specific to a functional area. Record-oriented queries are described in other chapters. Display-oriented queries are described here.

Each triggering event is listed below, with the applicable form of the message exchange.

2.16.12.1 A query is made for immediate response (Trigger Event Q01)

QRY	Query Message	Chapter
MSH	Message Header	2
QRD	Query Definition	2
[QRF]	Query Filter	2
[DSC]	Continuation Pointer	2

DSR	Display Response Message	Chapter
MSH	Message Header	2
MSA	Message Acknowledgement	2
[ERR]	Error	2
QRD	Query Definition	2
[QRF]	Query Filter	2
{ DSP }	Display Data	2
[DSC]	Continuation Pointer	2

The QRF and QRD segments from the QRY are echoed back in the response. The DSC segment contains the continuation pointer, if it is not null (*DSC-1-continuation pointer*).

2.16.12.1.1 Display query variants

If a display query has more than a single type of response (i.e., a DSR message with a different meaning, requiring different processing on the querying system), the second component of the Message Type field of the MSH segment may be used to indicate the response event type. For example, an ancillary name search display query may be defined using the query event code of DNM. The display response to such a query may be either a list of name matches (response event type is DNM) or the ancillary's display results for an exact match to the name query (response event type is NRS). See *table 0003 - event type code* and field notes for *MSH-9-message type*.

2.16.12.2 Deferred access

For clarity, A is the system initiating the query and B is the system sending the responses. Multiple queries and responses are permitted within single messages. The responses to a given query may be broken into several separate DSR messages. A single DSR message may contain responses to more than one QRY.

2.16.12.2.1 A query is sent for deferred response (trigger event Q02)

QRY (A to B)	Query Message	Chapter
MSH	Message Header	2
QRD	Query Definition	2
[QRF]	Query Filter	2
[DSC]	Continuation Pointer	2

ACK (B to A)	General Acknowledgement	Chapter
MSH	Message Header	2
MSA	Message Acknowledgement	2
[ERR]	Error	2

2.16.12.2.2 Deferred response to a query (trigger event Q03)

Later, perhaps more than once.

DSR (B to A)	Display Response Message	Chapter
MSH	Message Header	2
[MSA]	Message Acknowledgement	2
QRD	Query Definition	2
[QRF]	Query Filter	2
{ DSP }	Display Data	2
[DSC]	Continuation Pointer	2

ACK (A to B)	General Acknowledgement	Chapter
MSH	Message Header	2
MSA	Message Acknowledgement	2
[ERR]	Error	2

2.16.13 Query message implementation considerations

a)The particular allowable values for the filters in the QRD and QRF segments are determined among cooperating applications during implementation.

b)The format chosen for the query segments are very general. This might be read by prospective implementors to imply that the requirement for using the Standard is the ability to respond to a wide variety of inquiries. This is not the intent. The format here can be used with specific restrictions in any interface.

2.17 SPECIAL HL7 PROTOCOLS

This section contains several extensions to the basic HL7 message protocol. These extensions represent implementation choices, and are to be used on a site-specific and application-specific basis as needed.

2.18.1 Sequence number protocol

For certain types of data transactions between systems the issue of keeping databases synchronized is critical. An example is an ancillary system such as lab, which needs to know the locations of all inpatients to route stat results correctly. If the lab receives an ADT transaction out of sequence, the census/location information may be incorrect. Although it is true that a simple one-to-one acknowledgement scheme can prevent out-of-sequence transactions between any two systems, only the use of sequence numbers can prevent duplicate transactions.

Note: Although this sequence number protocol is limited to the use of sequence numbers on a single transaction stream between two applications, this sequencing protocol is sufficiently robust to allow the design of HL7-compatible store-and-forward applications.

a)definitions, initial conditions:

1)the system receiving the data stream is expected to store the sequence number of the most recently accepted transaction in a secure fashion before acknowledging that transaction. This stored sequence number allows comparison with the next transaction's sequence number, and the implementation of fault-tolerant restart capabilities.

2)the initiating system keeps a queue of outgoing transactions indexed by the sequence number. The length of this queue must be negotiated as part of the design process for a given link. The minimum length for this queue is one.

3)the sequence number is a positive (non-zero) integer; and it is incremented by one (by the initiating system) for each successive transaction.

b)starting the link:

1)the value of 0 (zero) for a sequence number is reserved: it is allowed only when the initiating system (re-)starts the link.

2)if the receiving system gets a transaction with a 0 (zero) in the sequence number field, it should respond with a general acknowledgement message whose MSA contains a sequence number one greater than the sequence number of the last transaction it accepted in the Expected Sequence Number field. If this value does not exist (as on the first startup of a given link), the MSA should contain a sequence number of -1, meaning that the receiving system will use the positive, non-zero sequence number of the first transaction it accepts as its initial sequence number (see resynching the link, item **e** below).

3)the initiating system then sends the transaction indexed by the expected sequence number (if that expected transaction is still on its queue). Otherwise the link is frozen until an operator intervenes.

c)normal operation of the link:

As it accepts each transaction, the receiving system securely stores the sequence number (which agrees with its expected sequence number), and then acknowledges the message by echoing the sequence number in *MSA-4-expected sequence number*.

d)error conditions (from point of view of initiating system). These are generated by the receiving system, by its comparison of the sequence number sent out (with the MSH in *MSH-13-sequence number*) with the expected sequence number (*MSA-4-expected sequence number* received with the MSA).

1)**expected sequence number is one greater than current value**. The previous acknowledgement was lost. That transaction was sent again. Correct by sending next transaction.

2)**expected sequence number less than current value**. Initiating system can try starting again by issuing a transaction with a sequence number of zero; or freeze the link for operator intervention.

3)**other errors**: freeze the link for operator intervention

e)forcing resynchronization of sequence numbers across the link. The value of -1 for a sequence number is reserved: it is allowed only when the initiating system is resynching the link. Thus if the receiving system gets a value of -1 in the sequence number field, it should return a general acknowledgement message with a -1 in the expected sequence number field. The receiving system then resets its sequence number, using the non-zero positive sequence number of the next transaction it accepts.

f)notes

When the initiating system sends a message with a sequence number of **0** or **-1** (see b or e above), the segments beyond the MSH need not be present in the message, or, if present, all fields can be null. In terms of the responding system, for these two cases, only a General Acknowledgement message is needed.

2.18.3 Continuation messages and segments

See Query Section 2.8.4 for a discussion of the continuation pointer segment and the continuation pointer field, and their use in the continuation of responses to queries and in the continuation of unsolicited update messages.

Besides the need to continue a message, there are occasional implementation conditions that force the continuation of a segment across messages. Such continued segments require the use of the ADD segment as follows:

a)the segment being continued (call it **ANY** for this example) is ended at an arbitrary character position and terminated with the Standard segment terminator (carriage return).

b)the following segment is the ADD segment. When it follows a segment being continued, the ADD segment contains no fields. Whether the message being continued is a response to a query, or an unsolicited update, the receiving system will use the continuation pointer (with the ADD segment) to continue the message.

c)when a response (to a query) is continued, the first segment after the QRD and QRF (on a continued query) will be the ADD segment. All the fields after the ADD segment identifier (fields 1-N) will be the continuation of the ANY segment. The receiving system will then use the continuation pointer to join the two parts of the ANY segment and continue processing the message.

d)for the continuation of an unsolicited update message, the ADD segment will be the first segment after the MSH segment. The receiving system will use the continuation pointer field in the MSH segment to identify the message being continued, and then use the ADD segment as described in c) to join the two parts of the ANY segment.

e)limitations: MSH, MSA, DSC, PID, QRD, QRF, URD and URS segments cannot be continued.

f)although the UU example given below is for a display message, there is nothing in the protocol to prevent a record-oriented UU from being continued in this fashion. In the unsolicited display message, the ADD record on the continuation comes just after the URD/[URS] pair instead of directly after the MSH.

g) transaction flow for a continued query-response pair with an ADD segment:

1) first query and response:

```
MSH
QRD
[QRF]

MSH
MSA
[ ERR ]
QRD
[QRF]
   {  DSP  }(last DSP segment is incomplete)
ADD          (ADD segment contains no fields)
DSC
```

2) second query and response:

```
MSH
QRD
[QRF]
DSC

MSH
MSA
[ ERR ]
QRD
[QRF]
ADD  (contains the remainder of the last DSP segment of the previous response)
  {DSP}(remaining DSP segments are complete)
```

> **Note:** This second response could itself be continued with a second DSC and (if needed) a second ADD segment prior to it.

f) transaction flow for a continued unsolicited message with a continued segment.

1)first unsolicited message and acknowledgement:

```
MSH
URD
[ URS ]
  {DSP}    (last DSP is incomplete)
ADD              (contains no fields)
DSC        (Continuation segment)

MSH        (General Acknowledgement)
MSA
[ ERR ]
```

2)second unsolicited message and acknowledgement:

```
MSH (contains continuation pointer from DSC segment
          of prior message)
ADD  (contains remainder of data from continued DSP
     segment from prior message)
{DSP}
```

> **Note:** This second message could itself be continued with a second DSC and (if needed) a second ADD segment prior to it.

```
MSH        (General Acknowledgement)
      MSA
```

```
[ ERR ]
```

2.18.5 HL7 batch protocol

There are instances when it is convenient to transfer a batch of HL7 messages. Common examples would be a batch of financial posting detail transactions (DFT's) sent from an ancillary to a financial system. Such a batch could be sent online using a common file transfer protocol, or offline via tape or diskette.

2.18.6.1 HL7 batch file structure

The structure of an HL7 batch file is given by the following (using the HL7 abstract message syntax defined in Section 2.4.8b):

```
[FHS]          (file header segment)
{ [BHS]        (batch header segment)
 { [MSH        (zero or more HL7 messages)
      ....
      ....
      ....
] }
  [BTS]        (batch trailer segment)
}
[FTS]          (file trailer segment)
```

Notes:

The sequence numbering protocol has a natural application in batch transfers. See the discussion of batch acknowledgements that follows.

Although a batch will usually consist of a single type of message, there is nothing in the definition that restricts a batch to only one message type.

The HL7 file and batch header and trailer segments are defined in exactly the same manner as the HL7 message segments. Hence the HL7 message construction rules of Section 2.4.7 can be used to encode and decode HL7 batch files.

There are only two cases in which an HL7 batch file may contain zero HL7 messages:

a)a batch containing zero HL7 messages may be sent to meet a requirement for periodic submission of batches when there are no messages to send.

b)a batch containing zero negative acknowledgement messages may be sent to indicate that all the HL7 messages contained in the batch being acknowledged are implicity acknowledged. See Section 2.9.3.3 c below.

2.18.6.2 Related segments and data usage

The following segments defined in Section 2.10 relate to the HL7 Batch Protocol:

BHS	Batch Header
BTS	Batch Trailer
FHS	File Header
FTS	File Trailer

The BTS segment contains a field, *BTS-3-batch totals*, which may have one or more totals drawn from fields within the individual messages. The method for computing such totals will be determined on a site or application basis unless explicitly stated in a functional chapter.

2.18.6.3 Acknowledging batches

In general, the utility of sending batches of data is that the data is accepted all-at-once, with errors processed on an exception basis. However, it is a permissible application of HL7 to acknowledge all messages. Several options for acknowledgement are given and will be chosen on an application basis. In these cases, the sequence numbering protocol can be useful to the applications processing the batch.

The options are:

a)all messages are acknowledged in the response batch.

b)the receiving system prints some form of batch control report, which is then dealt with manually by personnel at the sending system. No acknowledgements are performed by the protocol software.

c)an automated acknowledgement batch is created containing acknowledgement messages only for those messages containing errors. In this mode an empty acknowledgement batch may be created (i.e., an HL7 batch file without any HL7 acknowledgement messages).

In each case where there is a response batch, its format is a batch of individual messages. Each individual message is in the format defined for an on-line response in the chapters. Consider, for example, a batch that might be constructed to respond to a batch of Detailed Financial Transactions (Chapter 6). The messages in the response batch would consist entirely of ACK messages, since ACK is the response shown in Chapter 6.

When batches are retransmitted after the correction of errors, *BHS-12-reference batch control ID* should contain the batch control ID of the original batch.

2.18.6.4 Batch message as a query response

The HL7 query also can be used to query for a batch in the following manner:

a)use the value BB or BL of *QRD-5-deferred response type* to specify a batch response. The query will be acknowledged with a general acknowledgement as in the Deferred Access example above (see Section 2.8.6.2).

b)in addition, insert into the batch file the QRD and QRF segments as follows:

```
[FHS]              (file header segment)
 { [BHS]           (batch header segment)
   [QRD]           (the QRD and QRF define the
   [QRF]              query that this batch is a
                  response to)
```

```
        { MSH          (one or more HL7 messages)
          ....
          ....
          ....
        }

      [BTS]          (batch trailer segment)
    }
    [FTS]            (file trailer segment)
```

c)the acknowledgement of a batch is described in this chapter (see Section 2.9.3.3)

2.19 MESSAGE CONTROL SEGMENTS

The following segments are necessary to support the functionality described in this chapter.

2.20.1
2.10.1 MSH - message header segment

The MSH segment defines the intent, source, destination, and some specifics of the syntax of a message.

Figure 2-9 MSH attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	1	ST	R			00001	Field separator
2	4	ST	R			00002	Encoding characters
3	15	ST				00003	Sending application
4	20	ST				00004	Sending facility
5	30	ST				00005	Receiving application
6	30	ST				00006	Receiving facility
7	26	TS				00007	Date/time of message
8	40	ST				00008	Security
9	7	CM	R		0076	00009	Message type
10	20	ST	R			00010	Message control ID
11	1	ID	R		0103	00011	Processing ID
12	8	ID	R		0104	00012	Version ID
13	15	NM				00013	Sequence number
14	180	ST				00014	Continuation pointer
15	2	ID			0155	00015	Accept acknowledgement type
16	2	ID			0155	00016	Application acknowledgement type
17	2	ID				00017	Country code

2.10.1.0 MSH field definitions

2.20.2.1 Field separator (ST) 00001

Definition: actually the separator between the segment ID and the first real field, *MSH-2-encoding characters*. As such it serves as the separator and defines the character to be used as a separator for the rest of the message. Recommended value is **|**.

2.20.2.2 Encoding characters (ST) 00002

Definition: four characters in the following order: the component separator, repetition separator, escape character, and subcomponent separator. Recommended values are **^~\&**. See Section 2.4.4 - message delimiters.

2.20.2.3 Sending application (ST) 00003

Definition: available for interface with lower level protocols.

2.20.2.4 Sending facility (ST) 00004

Definition: addresses one of several occurrences of the same application within the sending system. Absent other considerations, the Medicare Provider ID might be used with an appropriate sub-identifier in the second component. Entirely site-defined.

2.20.2.5 Receiving application (ST) 00005

Definition: available for interface with lower level protocols.

2.20.2.6 Receiving facility (ST) 00006

Definition: identifies the receiving application among multiple identical instances of the application running on behalf of different organizations. See comments: sending facility.

2.20.2.7 Date/time of message (TS) 00007

Definition: date/time that the sending system created the message. If the time zone is specified, it will be used throughout the message as the default time zone.

2.20.2.8 Security (ST) 00008

Definition: in some applications of HL7 this field will be used to implement security features. Its use is not yet further specified.

2.20.2.9 Message type (CM) 00009

Components: <message type> ^ <trigger event>

Definition: first component is the message type edited by *table 0076 - message type*; second is the trigger event code edited by *table 0003 - event type code*. Receiving system uses this field to know the data segments to recognize, and possibly, the application to which to route this message. For certain queries, which may have more than a single response event type, the second component may, in the response message, vary to indicate the response event type. See the discussion of the display query variants in Section 2.8.6.1.1. The second component is not required on response or acknowledgement messages.

Table 0076 Message type

Value	Description	Owner	Chapter
ACK	General acknowledgement message	CNT	2
ADT	ADT message	ADT	3
ARD	Ancillary RPT (display)	ANR	7
BAR	Add/change billing account	BLN	6
DFT	Detail financial transaction	BLN	6
DSR	Display response	QRY	2
MCF	Delayed acknowledgement	CNT	2
ORF	Observ. result/record response	ANR	7
ORM	Order message	ORD	4
ORR	Order acknowledgement message	ORD	4
ORU	Observ result/unsolicited	ANR	7
OSQ	Order status query	ORD	4
RAR	Pharmacy administration information	ORD	4
RAS	Pharmacy administration message	ORD	4
RDE	Pharmacy encoded order message	ORD	4
RDR	Pharmacy dispense information	ORD	4
RDS	Pharmacy dispense message	ORD	4
RGV	Pharmacy give message	ORD	4
RGR	Pharmacy dose information	ORD	4
RER	Pharmacy encoded order information	ORD	4
ROR	Pharmacy prescription order response	ORD	4
RRA	Pharmacy administration acknowledgement	ORD	4
RRD	Pharmacy dispense acknowledgement	ORD	4
RRE	Pharmacy encoded order acknowledgement	ORD	4
RRG	Pharmacy give acknowledgement	ORD	4
QRY	Query	QRY	2
UDM	Unsolicited display message	QRY	2

2.20.2.10 Message control ID (ST) 00010

Definition: number or other identifier that uniquely identifies the message. The receiving system echoes this ID back to the sending system in the Message Acknowledgement segment (MSA).

2.20.2.11 Processing ID (ID) 00011

Definition: used to decide whether to process the message as defined in HL7 Application (level 7) Processing rules, above.

Table 0103 Processing ID

Value	Description
D	Debugging
P	Production
T	Training

2.20.2.12 Version ID (ID) 00012

Definition: matched by the receiving system to its own version to be sure the message will be interpreted correctly.

Table 0104 Version ID

Value	Description
2.0	Release 2.0 September 1988
2.0D	Demo 2.0 October 1988
2.1	Release 2.1 March 1990
2.2	Release 2.2 ? 1994

2.20.2.13 Sequence number (NM) 00013

Definition: non-null value in this field implies that the sequence number protocol is in use. This numeric field incremented by one for each subsequent value.

2.20.2.14 Continuation pointer (ST) 00014

Definition: used to define continuations in application-specific ways.

2.20.2.15 Accept acknowledgement type (ID) 00015

Definition: defines the conditions under which accept acknowledgements are required to be returned in response to this message. Required for enhanced acknowledgement mode. Refer to *table 0155 - acknowledgement conditions* for valid values.

2.20.2.16 Application acknowledgement type (ID) 00016

Definition: defines the conditions under which application acknowledgements are required to be returned in response to this message. Required for enhanced acknowledgement mode.

The following table contains the possible values for *MSH-15-accept acknowledgement type* and *MSH-16-application acknowledgement type*:

Table 0155 Accept/application acknowledgement conditions

Value	Description
AL	Always
NE	Never
ER	Error/reject conditions only
SU	Successful completion only

Note: If MSH-15 and MSH-16 are omitted (or are both null), the original Acknowledgement Mode rules are used.

2.20.2.17 Country code (ID) 00017

Definition: defines the country of origin for the message. It will be used primarily to specify default elements, such as currency denominations. ISO 3166 provides a list of country codes that may be used[2].

2.20.3
2.10.2 MSA - message acknowledgement segment

The MSA segment contains information sent while acknowledging another message.

Figure 2-10 MSA attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	2	ID	R		0008	00018	Acknowledgement Code
2	20	ST	R			00010	Message Control ID
3	80	ST				00020	Text Message
4	15	NM				00021	Expected Sequence Number
5	1	ID			0102	00022	Delayed Acknowledgement Type
6	100	CE				00023	Error Condition

2.10.2.0 MSA field definitions

2.20.4.1 Acknowledgement code (ID) 00018

Definition: see message processing rules.

Table 0008 Acknowledgement code

Value	Description
AA	Original mode: Application Accept Enhanced mode: Application Acknowledgement: Accept
AE	Original mode: Application Error Enhanced mode: Application Acknowledgement: Error
AR	Original mode: Application Reject Enhanced mode: Application Acknowledgement: Reject
CA	Enhanced mode: Accept Acknowledgement: Commit Accept
CE	Enhanced mode: Accept Acknowledgement: Commit Error
CR	Enhanced mode: Accept Acknowledgement: Commit Reject

2.20.4.2 Message control ID (ST) 00010

[2] Available from ISO 1 Rue de Varembe, Case Postale 56, CH 1211, Geneve, Switzerland

Definition: message control ID of the message sent by the sending system. It allows the sending system to associate this response with the message for which it is intended.

2.20.4.3 Text message (ST) 00020

Definition: optional text field that further describes an error condition. This text may be printed in error logs or presented to an end user.

2.20.4.4 Expected sequence number (NM) 00021

Definition: optional numeric field used in the sequence number protocol.

2.20.4.5 Delayed acknowledgement type (ID)00022

Definition: used only as described above, in Section 2.5.2. Otherwise this field is not used.

Table 0102 Delayed acknowledgement type

Value	Description
D	Message received, stored for later processing
F	Acknowledgement after processing

2.20.4.6 Error condition (CE) 00023

Components: <identifier>^<text>^<name of coding system>^<alternate identifier>^<alternate text>^<name of alternate coding system>

Definition: CE field allowing the acknowledging system to use a user-defined error code to further specify AR or AE type acknowledgements. This field is a generalized replacement for *MSA-3-text message*.

2.20.5 page * arabic532.10.3 ERR - error segment

The ERR segment is used to add error comments to acknowledgement messages.

Figure 2-11 ERR attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	80	CM	R	Y	0060	00024	Error Code and Location

2.10.3.0 ERR field definition

2.20.6.1 Error code and location (CM) 00024

Components: <segment ID (ST)> ^ <sequence (NM)> ^ <field position (NM)> ^ <code identifying error (CE)>

Definition: identifies an erroneous segment in another message. The second component is an index if there are more than one segment of type <segment ID>. For systems that do not use the HL7 Encoding Rules, the data item number may be used for the third component. The fourth component references a user-defined error table and is restricted from having any subcomponents as the subcomponent separator is now the CE's component separator.

2.20.7
2.10.4 QRD - query definition segment

The QRD segment is used to define a query.

Figure 2-12 QRD attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	26	TS	R			00025	Query Date/Time
2	1	ID	R		0106	00026	Query Format Code
3	1	ID	R		0091	00027	Query Priority
4	10	ST	R			00028	Query ID
5	1	ID			0107	00029	Deferred Response Type
6	26	TS				00030	Deferred Response Date/Time
7	10	CQ	R		0126	00031	Quantity Limited Request
8	20	ST	R	Y		00032	Who Subject Filter
9	3	ID	R	Y	0048	00033	What Subject Filter
10	20	ST	R	Y		00034	What Department Data Code
11	20	ST		Y		00035	What Data Code Value Qual.
12	1	ID			0108	00036	Query Results Level

2.10.4.0 QRD field definitions

2.20.8.1 Query date/time (TS) 00025

Definition: date the query was generated by the application program.

2.20.8.2 Query format code (ID) 00026

Definition: refer to *table 0106 - query format code* for valid codes.

Table 0106 Query format code

Value	Description
D	Response is in display format
R	Response is in record-oriented format

2.20.8.3 Query priority (ID) 00027

Definition: time frame in which the response is expected. Refer to *table 0091 - query priority* for valid codes. Table values and subsequent fields specify time frames for response.

Table 0091 Query priority

Value	Description
D	Deferred
I	Immediate

2.20.8.4 Query ID (ST) 00028

Definition: unique identifier for the query. Assigned by the querying application. Returned intact by the responding application.

2.20.8.5 Deferred response type (ID) 00029

Definition: refer to *table 0107 - deferred response type* for valid entries.

Table 0107 Deferred response type

Value	Description
B	Before the Date/Time specified
L	Later than the Date/Time specified

2.20.8.6 Deferred response date/time (TS) 00030

Definition: date/time before or after which to send a deferred response. If not present, the response can be sent when its available. (See *QRD-5-deferred response type* above).

2.20.8.7 Quantity limited request (CQ) 00031

Definition: maximum length of the response that can be accepted by the requesting system. Valid responses are numerical values given in the units specified in the second component. Refer to *table 0126 - quantity limited request* for valid entries. Default is LI lines.

Table 0126 Quantity limited request

Value	Description
CH	Characters
LI	Lines
PG	Pages
RD	Records
ZO	Locally defined

2.20.8.8 Who subject filter (ST) 00032

Definition: identifies the subject, or who the inquiry is about.

2.20.8.9 What subject filter (ID) 00033

Definition: describes the kind of information that is required to satisfy the request. Valid codes define the type of transaction inquiry and may be extended locally during implementation.

Table 0048 What subject filter

Value	Description
ADV	Advice/diagnosis
ANU	Nursing unit lookup (returns patients in beds, excluding empty beds)
APN	Patient name lookup
APP	Physician lookup
ARN	Nursing unit lookup (returns patients in beds, including empty beds)
APM	Medical record number query, returns visits for a medical record number
APA	Account number query, return matching visit
CAN	Cancel. Used to cancel a query
DEM	Demographics
FIN	Financial
MRI	Most recent inpatient
MRO	Most recent outpatient
NCK	Network clock
NSC	Network status change
NST	Network statistic
ORD	Order
OTH	Other
PRO	Procedure
RES	Result
RAR	Pharmacy administration information
RER	Pharmacy encoded order information
RDR	Pharmacy dispense information
RGR	Pharmacy give information
ROR	Pharmacy prescription information
STA	Status

See HL7 Implementation Guide for detailed examples of use of various query filter fields.

2.20.8.10 What department data code (ST) 00034

Definition: possible contents include test number, procedure number, drug code, item number, order number, etc. The contents of this field are determined by the contents of the previous field. This field could contain multiple occurrences separated by repetition delimiters.

2.20.8.11 What data code value qual (CM)00035

Components: <first data code value (ST)> ^ <last data code value (ST)>

Definition: what data code value qualifier. A window or range to further refine the inquiry. This field would contain start/stop separated by component separators.

2.20.8.12 Query results level (ID) 00036

Definition: used to control level of detail in results. Refer to *table 0108 - query results level* for valid codes. See chapters 4 and 7.

Table 0108 Query results level

Value	Description
O	Order plus order status
R	Results without bulk text
S	Status only
T	Full results

2.20.9
date \@ "MMMM d, yyyy"April 2, 2003
2.10.5 QRF - query filter segment

The QRF segment is used with the QRD segment to refine the content of a query further.

Figure 2-13 QRF attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	20	ST	R	Y		00037	Where Subject Filter
2	26	TS				00038	When Data Start Date/Time
3	26	TS				00039	When Data End Date/Time
4	20	ST		Y		00040	What User Qualifier
5	20	ST		Y		00041	Other QRY Subject Filter
6	12	ID		Y	0156	00042	Which Date/Time Qualifier
7	12	ID		Y	0157	00043	Which Date/Time Status Qualifier
8	12	ID		Y	0158	00044	Date/Time Selection Qualifier

2.10.5.0 QRF field definitions

2.20.10.1 Where subject filter (ST) 00037

Definition: identifies the department, system, or subsystem to which the query pertains. This field may repeat as in LAB~HEMO, etc.

2.20.10.2 When data start date/time (TS) 00038

Definition: data representing dates and times equal or after this value should be included.

2.20.10.3 When data end date/time (TS) 00039

Definition: data representing dates and times the same as or before this date should be included.

2.20.10.4 What user qualifier (ST) 00040

Definition: an identifier to further define characteristics of the data of interest.

2.20.10.5 Other QRY subject filter (ST) 00041

Definition: a filter defined locally for use between two systems. This filter uses codes and field definitions which have specific meaning only to the applications and/or site involved.

2.20.10.6 Which date/time qualifier (ID) 00042

Definition: specifies type of date referred to in *QRF-2-when data start date/time* and *QRF-3-when data end date/time*.

Table 0156 Which date/time qualifier

Value	Description
ORD	Order date/time
CAN	Cancellation date/time
SCHED	Schedule date/time
COL	Collection date/time, equivalent to film or sample collection date/time
RCT	Specimen receipt date/time, receipt of specimen in filling ancillary (Lab)
REP	Report date/time, report date/time at filiing ancillary (i.e., Lab)
ANY	Any date/time within a range

2.20.10.7 Which date/time status qualifier (ID) 00043

Definition: specifies status type of objects selected in date range defined by *QRF-2-when data start date/time* and *QRF-3-when data end date/time*).

Table 0157 Which date/time status qualifier

Value	Description
PRE	Preliminary
REP	Report completion date/time
CFN	Current final value, whether final or corrected
FIN	Final only (no corrections)
COR	Corrected only (no final with corrections)
ANY	Any status

2.20.10.8 Date/time selection qualifier (ID) 00044

Definition: allows specification of certain types of values within the date/time range.

Table 0158 Date/time selection qualifier

Value	Description
1ST	First value within range
ALL	All values within the range
LST	Last value within the range
REV	All values within the range returned in reverse chronological order (This is the

	default if not otherwise specified.)

2.20.11
2.10.6 URD - results/update definition segment

The URD segment is used in sending unsolicited updates about orders and results. It's purpose is similar to that of the QRD segment, but from the results/unsolicited update point of view. Some of the fields have parallels in the QRD segment.

Figure 2-14 URD attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	26	TS				00045	R/U Date/Time
2	1	ID			0109	00046	Report Priority
3	20	ST	R	Y		00047	R/U Who Subject Definition
4	3	ID		Y	0048	00048	R/U What Subject Definition
5	20	ST		Y		00049	R/U What Department Code
6	20	ST		Y		00050	R/U Display/Print Locations
7	1	ID			0108	00051	R/U Results Level

2.10.6.0 URD field definitions

2.20.12.1 R/U date/time (TS) 00045

Definition: date and time the update was generated by the application program.

2.20.12.2 Report priority (ID) 00046

Definition: priority associated with this report or update. Refer to *table 0109 - report priority* for valid codes.

Table 0109 Report priority

Value	Description
R	Routine
S	Stat

2.20.12.3 R/U who subject definition (ST) 00047

Definition: definition of the subject, or who the report is about.

2.20.12.4 R/U what subject definition (ID) 00048

Definition: describes the kind of information that is provided in the report. Valid codes are the type of transaction inquiry. Refer to *table 0048 - what subject filter* for valid codes.

This table may be extended by local agreement during implementation. See HL7 Implementation Guide for detailed examples of use of various query filter fields.

2.20.12.5 R/U what department code (ST) 00049

Definition: possible contents include test number, procedure number, drug code, item number, order number, etc. The contents of this field are determined by the contents of the previous field. This field could contain multiple occurrences separated by repetition delimiters.

2.20.12.6 R/U display/print locations (ST) 00050

Definition: a list of the locations to which the report should be distributed.

2.20.12.7 R/U results level (ID) 00051

Definition: used to control level of detail in results. Refer to *table 0108 - query results level* for valid codes. Default level is T for full results. See chapters 4 and 7.

2.20.13
2.10.7 URS - unsolicited selection segment

The URS segment is identical with the QRF segment, except that, if the name of any field contains **Query** (of QRY), this word has been changed to **Results** (See *URS-5-R/U other results subject definition*).

Figure 2-15 URS attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	20	ST	R	Y		00052	R/U Where Subject Definition
2	26	TS				00053	R/U When Data Start Date/Time
3	26	TS				00054	R/U When Data End Date/Time
4	20	ST		Y		00055	R/U What User Qualifier
5	20	ST		Y		00056	R/U Other Results Subject Definition
6	12	ID		Y	0156	00057	R/U Which Date/Time Qualifier
7	12	ID		Y	0157	00058	R/U Which Date/Time Status Qualifier
8	12	ID		Y	0158	00059	R/U Date/Time Selection Qualifier

2.10.7.0 URS field definitions

2.20.14.1 R/U where subject definition (ST) 00052

Definition: identifies the department, system, or subsystem to which the result pertains. This field may repeat as in **LAB~HEMO**, etc.

2.20.14.2 R/U when data start date/time (TS) 00053

Definition: date/time the result starts. (if applicable).

2.20.14.3 R/U when data end date/time (TS) 00054

Definition: date/time the result ends. (if applicable).

2.20.14.4 R/U what user qualifier (ST) 00055

Definition: an identifier to define further the characteristics of the data that are of interest.

2.20.14.5 R/U other results subject definition (ST) 00056

Definition: further qualifier defined locally for use between two systems. This filter uses codes and field definitions that have specific meaning only to the application and/or site involved.

2.20.14.6 R/U Which date/time qualifier (ID) 00057

Definition: specifies type of date referred to in *URS-2-when data start date/time* and *URS-3-when data end date/time*. Refer to *table 0156 - which date/time qualifier*.

2.20.14.7 R/U Which date/time status qualifier (ID) 00058

Definition: specifies status type of objects selected in date range defined by *URS-2-when data start date/time* and *URS-3-when data end date/time*. Refer to *table 0157 - date/time status qualifier*.

2.20.14.8 R/U Date/time selection qualifier (ID) 00059

Definition: allows specification of certain types of values within the date/time range. Refer to *table 0158 - date/time selection qualifier*.

2.20.15
2.10.8 DSC - Continuation pointer segment

The DSC segment is used in the continuation protocol.

Figure 2-16 DSC attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	180	ST				00060	Continuation Pointer

2.10.8.0 DSC field definition

2.20.16.1 Continuation pointer (ST) 00060

Definition: see description of Continuation Fields in Section 2.8.4. In an initial query, this field is null. If the responder returns a value of null or not present, then there is no more data to fulfill any future continuation requests. For use with continuations of unsolicited messages, see sections 2.7.2 and 2.9.2. Note that continuation protocols work with both display and record-oriented messages.

2.20.17
2.10.9 DSP - display data segment

The DSP segment is used to contain data that has been preformatted by the sender for display. The semantic content of the data is lost; the data is simply treated as lines of text.

Figure 2-17 DSP attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	4	SI				00061	Set ID - Display Data
2	4	SI				00062	Display Level
3	300	TX	R			00063	Data Line
4	2	ST				00064	Logical Break Point
5	20	TX				00065	Result ID

2.10.9.0 DSP field definitions

2.20.18.1 Set ID - display data (SI) 00061

Definition: used optionally to number multiple display segments.

2.20.18.2 Display level (SI) 00062

Definition: individual sites or applications may assign numbering to define groups of data elements.

2.20.18.3 Data line (TX) 00063

Definition: contains an actual line as it should be displayed. As described for the TX data type, highlighting and other special display characteristics may be included.

2.20.18.4 Logical break point (ST) 00064

Definition: non-null if this line is the last line of a logical break point in the response as defined by the responding system. See Section 2.8.5 for the discussion of Logical display break points.

2.20.18.5 Result ID (TX) 00065

Definition: if the user selects a result ID (defined by *DSP-5-logical break point*) from the screen display corresponding to a record in which *DSP-5-result ID* is non-null, the application can initiate a second query (a separate session) to the ancillary with the *QRD-10-what department data code* filled in with this non-null value (e.g., the ancillary accession number or its equivalent). The ancillary response will contain the report referenced by this result ID (e.g., accession number). The ancillary should correlate the result ID with *DSP-4-logical break point* as follows: If more than one line of text is sent per result, *DSP-5-result ID* should be only non-null for a DSP segment that contains a non-null *DSP-4-logical break point*. This field may be broken into components by local agreement. A common example might be to include placer order number, filler order number, and universal service identifier. Whenever such fields are used as components of the result ID, their components will be sent as subcomponents.

2.20.19
2.10.10 ADD - addendum segment

The ADD segment is used to define the continuation of the prior segment in a continuation message. See Section 2.9.2 for details.

Figure 2-18 ADD attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	64k	ST				00066	Addendum Continuation Pointer

2.10.10.0 ADD field definition

2.20.20.1 Addendum continuation pointer (ST) 00066

Definition: used to define the continuation of the prior segment in a continuation message. See text for details When the ADD is sent after the segment being continued, it contains no fields. It is only a marker that the previous segment is being continued in a subsequent message. Thus fields 1-N are not present. The sequence designation, 1-N, means the remainder of the fields in the segment being continued. These remainder-of-the-segment-being-continued fields are present only when the ADD is sent with a continuation message.

2.20.21

2.10.11 FHS - file header segment

The FHS segment is used to head a file (group of batches) as defined in Section 2.9.3.

Figure 2-19 FHS attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	1	ST	R			00067	File Field Separator
2	4	ST	R			00068	File Encoding Characters
3	15	ST				00069	File Sending Application
4	20	ST				00070	File Sending Facility
5	15	ST				00071	File Receiving Application
6	20	ST				00072	File Receiving Facility
7	26	TS				00073	File Creation Date/Time
8	40	ST				00074	File Security
9	20	ST				00075	File Name/ID
10	80	ST				00076	File Header Comment
11	20	ST				00077	File Control ID
12	20	ST				00078	Reference File Control ID

2.10.11.0 FHS field definitions

2.20.22.1 File field separator (ST) 00067

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.22.2 File encoding characters (ST) 00068

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.22.3 File sending application (ST) 00069

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.22.4 File sending facility (ST) 00070

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.22.5 File receiving application (ST) 00071

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.22.6 File receiving facility (ST) 00072

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.22.7 File creation date/time (TS) 00073

Definition: field has the same definition as the corresponding field in the MSH segment.

2.20.22.8 File security (ST) 00074

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.22.9 File name/ID (ST) 00075

Definition: can be used by the application processing file. Its use is not further specified.

2.20.22.10 File header comment (ST) 00076

Definition: free text field, the use of which is not further specified.

2.20.22.11 File control ID (ST) 00077

Definition: used to identify a particular file uniquely. It can be echoed back in *FHS-12-reference file control ID.*

2.20.22.12 Reference file control ID (ST) 00078

Definition: value of *FHS-11-file control ID* when this file was originally transmitted. Not present if this file is being transmitted for the first time.

2.20.23

2.10.12 FTS - file trailer segment

The FTS segment defines the end of a file.

Figure 2-20 FTS attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	10	NM				00079	File Batch Count
2	80	ST				00080	File Trailer Comment

2.10.12.0 FTS field definitions

2.20.24.1 File batch count (NM) 00079

Definition: contains the number of batches contained in this file.

2.20.24.2 File trailer comment (ST) 00080

Definition: free text field, the use of which is not further specified.

2.20.25 page * arabic802.10.13 BHS - batch header

The BHS segment defines the start of a batch.

Figure 2-21 BHS attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	1	ST	R			00081	Batch Field Separator
2	3	ST	R			00082	Batch Encoding Characters
3	15	ST				00083	Batch Sending Application
4	20	ST				00084	Batch Sending Facility
5	15	ST				00085	Batch Receiving Application
6	20	ST				00086	Batch Receiving Facility
7	26	TS				00087	Batch Creation Date/Time
8	40	ST				00088	Batch Security
9	20	ST				00089	Batch Name/ID/Type
10	80	ST				00090	Batch Comment
11	20	ST				00091	Batch Control ID
12	20	ST				00092	Reference Batch Control ID

2.10.13.0 BHS field definition

2.20.26.1 Batch field separator (ST) 00081

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.26.2 Batch encoding characters (ST) 00082

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.26.3 Batch sending application (ST) 00083

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.26.4 Batch sending facility (ST) 00084

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.26.5 Batch receiving application (ST) 00085

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.26.6 Batch receiving facility (ST) 00086

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.26.7 Batch creation date/time (TS) 00087

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.26.8 Batch security (ST) 00088

Definition: has the same definition as the corresponding field in the MSH segment.

2.20.26.9 Batch name/ID/type (ST) 00089

Definition: can be used by the application processing the batch. It can have extra components if needed.

2.20.26.10 Batch comment (ST) 00090

Definition: comment field that is not further defined in the HL7 protocol.

2.20.26.11 Batch control ID (ST) 00091

Definition: used to uniquely identify a particular batch. It can be echoed back in *BHS-12-reference batch control ID* if an answering batch is needed.

2.20.26.12 Reference batch control ID (ST) 00092

Definition: value of *BHS-11-batch control ID* when this batch was originally transmitted. Not present if this batch is being sent for the first time. See definition for *BHS-11-batch control ID*.

2.20.27

2.10.14 BTS - batch trailer segment

The BTS segment defines the end of a batch.

Figure 2-22 BTS attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	10	ST				00093	Batch Message Count
2	80	ST				00094	Batch Comment
3	100	CM		Y		00095	Batch Totals

2.10.14.0 BTS field definitions

2.20.28.1 Batch message count (ST) 00093

Definition: count of the individual messages contained within the batch.

2.20.28.2 Batch comment (ST) 00094

Definition: comment field that is not further defined in the HL7 protocol.

2.20.28.3 Batch totals (CM) 00095

Components: <total 1 (NM)> ^ <total 2 (NM)> ^ ...

Definition: as many types of totals as needed for the batch may be carried by this field as separate components. Each component is an NM data type.

2.20.29
next2.10.15 NTE - notes and comments segment

The NTE segment is defined here for inclusion in messages defined in other chapters. It is a common format for sending notes and comments.

Figure 2-23 NTE attributes

SEQ	LEN	DT	R/O	RP/#	TBL#	ITEM#	ELEMENT NAME
1	4	SI				00096	Set ID - Notes and Comments
2	8	ID			0105	00097	Source of Comment
3	64k	FT		Y		00098	Comment

2.10.15.0 NTE field definitions

2.20.30.1 Set ID - notes and comments (SI) 00096

Definition: may be used where multiple NTE segments are included in a message. Their numbering must be described in the application message definition.

2.20.30.2 Source of comment (ID) 00097

Definition: used when source of comment must be identified. This table may be extended locally during implementation.

Table 0105 Source of comment

Value	Description
L	Ancillary (filler) department is source of comment
P	Orderer (placer) is source of comment
O	Other system is source of comment

2.20.30.3 Comment (FT) 00098

Definition: comment contained in the segment.

Note: In the current HL7 version, this is an FT rather than a TX data type. Since there is no difference between an FT data type without any embedded formatting commands, and a TX data type, this change is compatible with the previous version.

2.20.31
2.10.16 Miscellaneous HL7 tables used across all chapters.

2.20.32.1 Yes/No indicator table

Table 0136 Y/N indicator

Value	Description
Y	Yes
N	No

2.21
2.11 SAMPLE CONTROL AND QUERY MESSAGES

2.22.1 General Acknowledgement

LAB acknowledges the message that ADT sent identified as ZZ9380. (LAB and ADT, the sending and receiving system IDs, are site-defined.) Both systems are associated with the same FACILITY, 767543. There is no trigger event for an acknowledgement, so the second component of the MESSAGE TYPE field is not present. The component separator may be present there, but need not be. The AA code in the MSA segment indicates that the message was accepted by the application.

```
MSH|^~\&|LAB|767543|ADT|767543|19900314130405||ACK^|XX3657|P|2.1<CR>
MSA|AA|ZZ9380<CR>
```

2.22.3 Error return

The AR code in MSA indicates that the application rejected the message for functional reasons. The optional ERR segment includes here that the 16th field of the PID segment with the SET ID value of 1 had an error which was defined by the locally-established code X3L. The optional text message UNKNOWN COUNTY CODE in the link is designed to help programmers and support personnel while reviewing message logs.

```
MSH|^~\&|LAB|767543|ADT|767543|199003141304-0500||ACK^|XX3657|P|2.1<CR>
MSA|AR|ZZ9380|UNKNOWN COUNTY CODE<CR>
ERR|PID^1^16^X3L<CR>
```

2.22.5 Sequence number: initial message

The sender initiates the link with a message that has no functional content. The sequence number is 0. The message type and event code are not used.

```
MSH|^~\&|ADT|767543|LAB|767543|199003141304-0500||^|XX3657|P|2.1|0<CR>
```

The responder uses a general acknowledgement. The expected sequence number is 1.

```
MSH|^~\&|LAB|767543|ADT|767543|199003141304-0500||ACK^|ZZ9380|P|2.1<CR>
MSA|AA|XX3657||1<CR>
```

2.22.7 Query with display-oriented response

Query for all lab results on patient #12233. The query is made at 11:00 a.m., 9/11/87. The Query anticipates an immediate display-oriented response.

```
MSH|^~\&|ICU||LAB01|||QRY^Q01|MSG00001|P|2.1<CR>
QRD|198709111012|D|I|4387|||20^LI|12233|RES|ALL<CR>
```

The response to the above query might look like the following:

```
MSH|^~\&|LAB01||ICU|||DSR|ZXT23461|P|2.1<CR>
MSA|AA|MSG00001P<CR>
QRD|198709111012|D|I|4387|||20^LI|12233|RES|ALL<CR>
DSP|||RESULTS FOR PATIENT#12233   SMITH, JOHN H. 09/11/87<CR>
DSP|||SPECIMEN#H85 COLLECTED 09/11/87 /07/0/0<CR>
DSP<CR>
DSP|||ELECTROLYTES<CR>
DSP|||     SODIUM           140   [135-148]   MEQ/L     STAT<CR>
DSP|||     POTASSIUM        4.0   [3.5-5.0]   MEQ/L     STAT<CR>
```

```
DSP|||      CHLORIDE          89    [95-111]    MEQ/L      STAT<CR>
DSP|||      CO2               20    [20-30]           MEQ/L      STAT<CR>
DSP||||LB<CR>
DSP|||CBC<CR>
DSP|||      HEMOGLOBIN              [13.5-18.0]<CR>
DSP|||      HEMATOCRIT        45    [40-54]           %<CR>
DSP|||      RED CELL COUNT    5.0   [4.6-6.2]   M/MM3<CR>
DSP|||      MCHC              32    [32-36]           G/DL<CR>
DSP|||      MCH               28    [26-32]           PG<CR>
DSP|||      MCV               85    [81-101]    FL<CR>
DSP|||      WHITE CELL CNT    7.5   [5.0-10.0]  K/MM3<CR>
DSP||||LB<CR>
DSP|||SPECIMEN#B24   COLLECTED 9/10/87<CR>
DSC|12333H85;12<CR>
```

A continuation query would echo back the contents of the Continuation Pointer in the DSC segment:

```
MSH|^~\&|ICU||LAB01|||QRY^Q01|MSG00003|P|2.1<CR>
QRD|198709111012|D|I|4387|||20^LI|12233|RES|ALL<CR>
DSC|12333H85;12<CR>
```

This response shows that there is no further data by leaving the Continuation Pointer not present. This could be done by sending the DSC segment ID with no data, but the example does the same thing by totally omitting the DSC segment.

```
MSH|^~\&|LAB01||ICU|||DSR|ZXT23469|P|2.1<CR>
MSA|AA|MSG00003|<CR>
QRD|198709111012|D|I|4387|||20^LI|12233|RES|ALL<CR>
DSP|||RESULTS FOR PATIENT#12233   SMITH, JOHN H. 09/11/87<CR>
DSP|||SPECIMEN#H85 COLLECTED 09/10/87 /07/0/0<CR>
DSP<CR>
DSP|||ELECTROLYTES<CR>
DSP|||      SODIUM       136   [135-148]   MEQ/L      STAT<CR>
DSP|||      POTASSIUM    4.2   [3.5-5.0]   MEQ/L      STAT<CR>
DSP|||      CHLORIDE     91    [95-111]    MEQ/L      STAT<CR>
DSP|||      CO2          25    [20-30]           MEQ/L      STAT<CR>
DSP||||LB<CR>
```

2.22.9Master file update examples: with original and enhanced acknowledgement protocol

This example shows the lab system using the Master Files specification to send two update test dictionary entries to an ICU system. The ODC (observation dictionary) segment, currently under development by HL7 and ASTM, carries the dictionary information. Several varieties of acknowledgement are shown. The choice of acknowledgement mode is site-specific.

2.22.10.1 Original mode example:

```
MSH|^~\&|LABxxx|ClinLAB|ICU||19910918060544||MFN^M03|MSGID002|P|2.2
```

```
MFI|LABxxx^Lab Test Dictionary^L|UPD|||AL
MFE|MUP|199109051000|199110010000|12345^WBC^L
OM1|...
MFE|MUP|199109051015|199110010000|6789^RBC^L
OM1|...
```

Original mode acknowledgement of the HL7 message according to MFI Response Level Code of AL.

```
MSH|^~\&|ICU||LABxxx|ClinLAB|19910918060545||MFK|MSGID99002|P|2.2
MSA|AA|MSGID002
MFI|LABxxx^Lab Test Dictionary^L|UPD|||MFAA
MFA|MUP|199110010000|199110010040|S|12345^WBC^L
MFA|MUP|199110010000|199110010041|S|6789^RBC^L
```

2.22.10.2 Enhanced mode example

2.22.10.2.1 Initial Message with accept acknowledgement

```
MSH|^~\&|LABxxx|ClinLAB|ICU||19910918060544||MFN^M03|MSGID002|P|2.2|||AL|AL
MFI|LABxxx^Lab Test Dictionary^L|UPD|||AL
MFE|MUP|199109051000|199110010000|12345^WBC^L
OM1|...
MFE|MUP|199109051015|199110010000|6789^RBC^L
OM1|...

MSH|^~\&|ICU||LABxxx|ClinLAB|19910918060545||MSA|MSGID99002|P|2.2
MSA|CA|MSGID002
```

2.22.10.2.2 Application acknowledgement message

```
MSH|^~\&|ICU||LABxxx|ClinLAB|19911001080504||MFK|MSGID5002|P|2.2|||AL|
MSA|AA|MSGID002
MFI|LABxxx^Lab Test Dictionary^L|UPD|||MFAA
MFA|MUP|199109051000|199110010040|S|12345^WBC^L
MFA|MUP|199109051015|199110010041|S|6789^RBC^L

MSH|^~\&|LABxxx|ClinLAB|ICU||19911001080507||ACK|MSGID444|P|2.2
MSA|CA|MSGID5002
```

2.22.10.3 Delayed application acknowledgement

Note: If the MFN message of 2.11.5.1 had not required an application acknowledgement at the message level (i.e., the application acknowledgement code of the MSH segment = NE), the (Master Files Chapter defined) MFD message could be used to provide a delayed application level acknowledgement not tied to the original MFN message.

The following example includes an acknowledgement for an MFE segment not in the original message. This additional MFE was sent via another MFN message.

2.22.10.3.1 Initial message with accept acknowledgement

```
MSH|^~\&|LABxxx|ClinLAB|ICU||19910918060544||MFN^M03|MSGID002|P|2.2|||AL|NE
MFI|LABxxx^Lab Test Dictionary^L|UPD|||AL
MFE|MUP|199109051000|199110010000|12345^WBC^L
OM1|...
MFE|MUP|199109051015|199110010000|6789^RBC^L
OM1|...

MSH|^~\&|ICU||LABxxx|ClinLAB|19910918060545||MSA|MSGID99002|P|2.2
MSA|CA|MSGID002
```

2.22.10.3.2 Delayed application acknowledgement

```
MSH|^~\&|ICU||LABxxx|ClinLAB|19911001080504||MFD|MSGID65002|P|2.2|||AL|
MFI|LABxxx^Lab Test Dictionary^L|UPD|||MFAA
MFA|MUP|199109051000|199110010040|S|12345^WBC^L
MFA|MUP|199109051015|199110010041|S|6789^RBC^L
MFA|MUP|199109051025|199110010041|S|4339^HGB^L

MSH|^~\&|LABxxx|ClinLAB|ICU||19911001080507||ACK|MSGID444|P|2.2
MSA|CA|MSGID65002
```

2.23 OUTSTANDING ISSUES

The following items are being discussed in the Control/Query technical committee for addition to future versions of HL7.

1. Inclusion of alternate character sets within HL7 messages. HL7 intends to allow the use of alternate character sets in a manner that is consistent with other US standards groups (such as ASTM E1238-94) and with various international standards groups (such as CEN/TC-251).

2. Investigate extending the HL7 query paradigm to include a subset of SQL defined on an implicit table structure consistent with a subset of current HL7 segment definitions.

3. Rationalization and clarification of event structures.

4. Creation of a network server for HL7 tables so that updates to them can be made public immediately, rather than waiting until the publication of the next version of the standard.

5. Extensions to the encoding rules for Version 3.