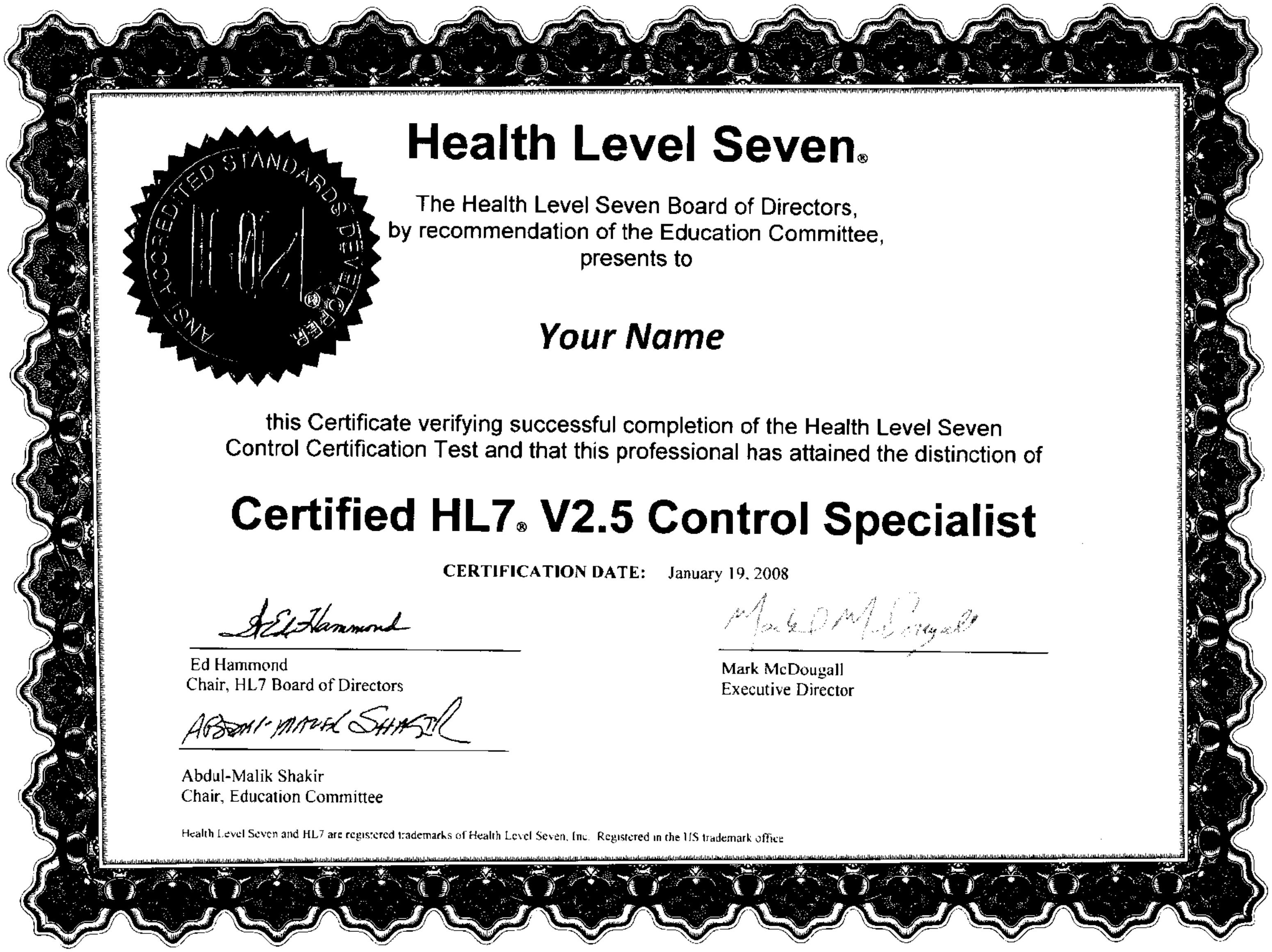
Introduction to HL7 Messaging Specifications for Data Interchange

June 2013

To Be Achieved



Health Level Seven®

The Health Level Seven Board of Directors,
by recommendation of the Education Committee,
presents to

Your Name

this Certificate verifying successful completion of the Health Level Seven Control Certification Test and that this professional has attained the distinction of

Certified HL7® V2.5 Control Specialist

CERTIFICATION DATE: January 19, 2008

[illegible]
Ed Hammond
Chair, HL7 Board of Directors

[illegible]
Mark McDougall
Executive Director

[illegible]
Abdul-Malik Shakir
Chair, Education Committee

Health Level Seven and HL7 are registered trademarks of Health Level Seven, Inc. Registered in the US trademark office

CitiusTech

Agenda

Why HL7 ?

- Hospitals and other healthcare provider organizations typically have many different computer systems used for everything from billing records to patient tracking.
- All of these systems should communicate with each other (or "interface") when they receive new information but not all do so.
- HL7 specifies a number of flexible standards, guidelines, and methodologies by which various healthcare systems can communicate with each other.
- Such guidelines or data standards are a set of rules that allow information to be shared and processed in a uniform and consistent manner.
- These data standards are meant to allow healthcare organizations to easily share clinical information.
- Theoretically, this ability to exchange information should help to minimize the tendency for medical care to be geographically isolated and highly variable

- Video on How HL7 Work

Introduction (1/8)

Information Systems in the healthcare enterprise

- Appointment schedulers
- Patient Registration Systems
- Service “order filler” systems, e.g.
  - Laboratory
  - Radiology
  - Surgery
- Billing Systems
- Others

Introduction (2/8)

What these information systems share?

- Patient demographic data
- Patient healthcare history
- Practitioner Information
- Hospital and clinical information
- Other common data

Introduction (3/8)

How Information Systems Communicate?

- Batch Processing
  - Update files
  - Tapes
- Real Time
  - Messages

Introduction (4/8)

Using Messages in Real-time communication

- List the data to be carried in the message
  - Patient data
  - Visit data
  - Doctor data

- Define a format for carrying the data
  - Defined by the enterprise
  - Defined by the system manufacturer

    Or
  - Standards based`

Introduction (5/8)

Where does HL7 enter the picture?

- Provides a set of predefined logical formats
  - For messages
  - For elements of the messages
- Assists in physical message formatting
  - Provides the HL7 default formatting also known as Encoding Rules/7 or ER/7
  - Gives guidance for using other formats such as XML
- Provides guidelines for communication
  - Lower layer protocols information
  - Acknowledgement protocols
- Facilitates communication in healthcare settings by providing standards for exchange of data among healthcare applications

Introduction (6/8)

What does HL7 Mean?

- Operates at the application level, which is 7th layer of OSI model
- Defines the sending application, receiving application, and the message being exchanged between them
- Makes no assumption about underlying lower level protocols (network, transport etc.)

Introduction (7/8)

Where does HL7 Operate?

- Patient Administration
- Order Entry
- Financial Management
- Observation Reporting
- Master files and Indexes
- Medical Records / Information Management
- Scheduling and Logistics
- Patient Care
- Patient Referral

Introduction (8/8)

- Makes no assumption about the ultimate use of data.
- Makes no assumptions about the ownership of data.
- Makes no assumptions 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 Versions (1/3)

- HL7 develops conceptual standards (e.g., HL7 RIM), document standards (e.g., HL7 CDA), application standards (e.g., HL7 CCOW), and messaging standards (e.g., HL7 v2.x and v3.0).
- Messaging standards are particularly important because they define how information is packaged and communicated from one party to another.
- Such standards set the language, structure and data types required for seamless integration from one system to another

HL7 Versions – V2.x (2/3)

- The HL7 version 2 standard has the aim to support hospital workflows.
- It was originally created in 1989.
- HL7 version 2 defines a series of electronic messages to support administrative, logistical, financial as well as clinical processes.
- Since 1987 the standard has been updated regularly, resulting in versions 2.1, 2.2, 2.3, 2.3.1, 2.4, 2.5, 2.5.1 and 2.6.
- The v2.x standards are backward compatible (i.e. a message based on version 2.3 will be understood by an application that supports version 2.6).
- HL7 v2.x messages use a human-readable (ASCII), non-XML encoding syntax based on segments (lines) and one-character delimiters.
- HL7 v2.x has allowed for the interoperability between electronic Patient Administration Systems (PAS), Electronic Practice Management (EPM) systems, Laboratory Information Systems (LIS), Dietary, Pharmacy and Billing systems as well as Electronic Medical Record (EMR) or Electronic Health Record (EHR) systems.
- Currently, HL7's v2.x messaging standard is supported by every major medical information systems vendor in the United States.

HL7 Versions – V3 (3/3)

- The HL7 version 3 standard has the aim to support all healthcare workflows.
- The v3 standard, as opposed to version 2, is based on a formal methodology and object-oriented principles.
- The Reference Information Model (RIM) is the cornerstone of the HL7 Version 3 development process and an essential part of the HL7 V3 development methodology.
- RIM expresses the data content needed in a specific clinical or administrative context and provides an explicit representation of them in form of classes and connections that exist between the information carried in the fields of HL7 messages.
- The HL7 version 3 messaging standard defines a series of electronic messages (called *interactions*) to support all healthcare workflows.
- HL7 v3 messages are based on an XML encoding syntax.
- The HL7 version 3 Clinical Document Architecture (CDA) is an XML-based mark-up standard intended to specify the encoding, structure and semantics of clinical documents for exchange.

Agenda

- Introduction
- **Conceptual Approach**
- Communications Environment
- Message Framework
- Message Construction Rules
- Use of Escape Sequences
- Version Compatibility Definition
- Acknowledgment Messages
- Message Control Segments

Conceptual Approach (1/3)

Trigger Event

- An event in the real world of healthcare creates the need for data to flow among systems

Rules for trigger events

- Each trigger event is associated with only one message type.
- A message type may have multiple trigger events associated with it.

Conceptual Approach (2/3)

Models of data flow

- Declarative Model: Unsolicited update/acknowledgement
- Interrogative Model: Query/Response
- Imperative Model: Operational Request/Confirmation

Queries/ Result

- for data regarding a single patient, e.g., send all lab results for patient #123456
- for data regarding multiple patients, e.g., send the list of patients whose attending physician is Dr. #123
- for data that is not patient related, e.g., send the age specific normal values for serum protein.

Conceptual Approach (3/3)

Unsolicited update/Acknowledgement

When the transfer of information is initiated by the application system that deals with the triggering event, the transaction is termed as an **unsolicited update.**

For e.g. ADT/ACK - admit/visit notification (event A01). An A01 event can be used to notify:

- Nursing system that the patient has been admitted and needs a care plan prepared
- The pharmacy system that a patient has been admitted and may be legitimately prescribed drugs.
- The finance system of the start of the billing period
- The dietary system that a new patient has been installed and requires dietary services.

Agenda

Communication Environment (1/2)

The universe of environments of interest to HL7 includes:

- ad hoc environments that do not provide even basic transport reliability.
    - e.g. point-to-point RS-232 links, modems

- environments that support a robust transport level, but do not meet the high level requirements.
    - e.g. TCP/IP, DECNET, and SNA.

- ISO and proprietary networks that implement up to presentation and other high level services.
    - e.g. IBM’s SNA LU6.2 and SUN Microsystems’s NFS

- two or more applications running on the same physical and/or logical machine that are not tightly integrated.
    - e.g., Pipes in a UNIX System

Communication Environment (2/2)

The HL7 Standard assumes that the communications environment will provide the following capabilities

- **Error free transmission**
  Applications can assume that they correctly received all of the transmitted bytes in the order in which they were sent. However, sending applications may not assume that the message was actually received without receiving an acknowledgment message.

- **Character conversion**
  If the two machines exchanging data use different representations of the same character set, the communications environment will convert the data from one representation to the other.

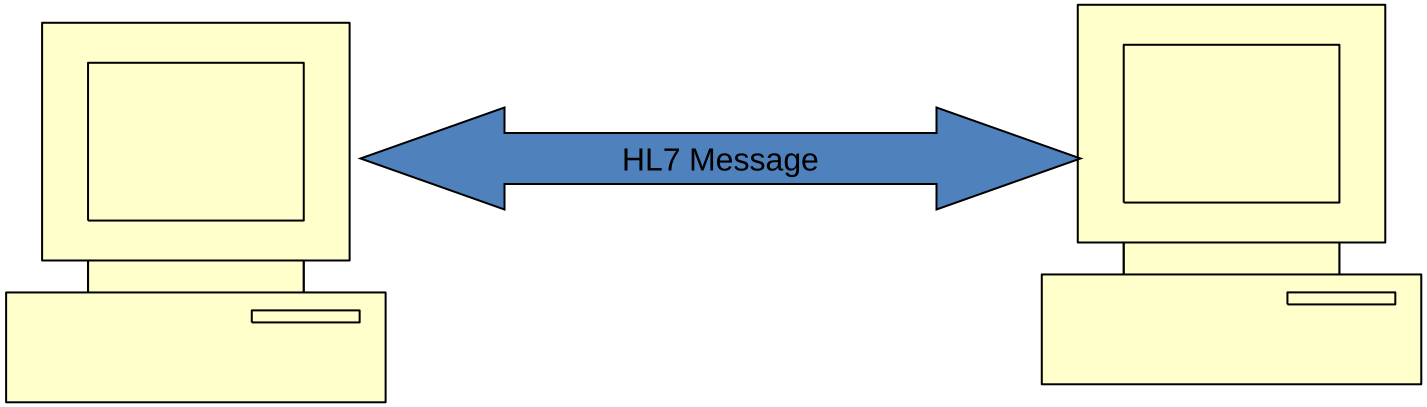
- **Message length**
  HL7 sets no limits on the maximum size of HL7 messages. The Standard assumes that the communications environment can transport messages of any length that might be necessary. In practice, sites may agree to place some upper bound on the size of messages and may use the message continuation protocol.

Agenda

- Introduction
- Conceptual Approach
- Communications Environment
- **Message Framework**
- Message Construction Rules
- Use of Escape Sequences
- Version Compatibility Definition
- Acknowledgment Messages
- Message Control Segments

Message Framework (1/9)

What is a HL7 message?



Message : Atomic unit of data transferred between systems

Message Framework (2/9)

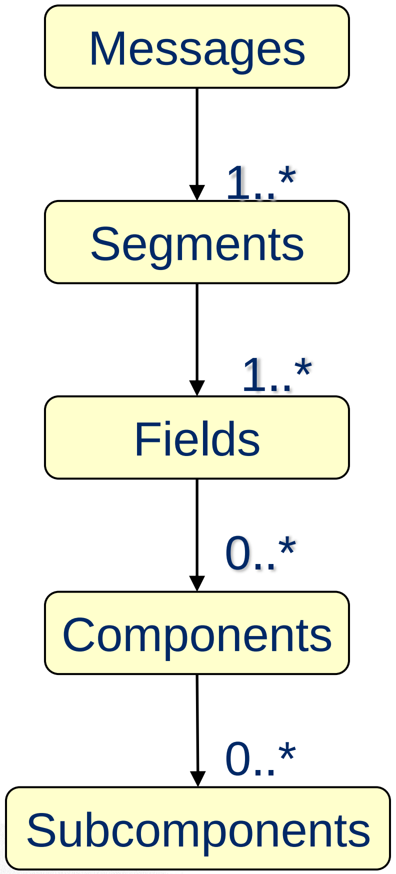
Example Patient Registration Message

```
MSH|^~\&#|MegaReg|UABHospC|ImgOrdMgr|UABImgCtr|20010529090131-
0500||ADT^A01|01052901|P|2.7

EVN|A01|200105290901||||200105290900

PID|||56782445^^^UAReg^PI~999855750^^^USSSA^SS||KLEINSAMPLE^BARRY^Q^JR#|
|19620910|M||C|260 GOODWIN CREST
DRIVE^^BIRMINGHAM^AL^35209^^H|||||||0105I30001

PV1||I|W^389^1^UABH^^^^3||||12345^MORGAN^REX^J^^^MD^^^UAMC^L||67890^
GRAINGER^LUCY^X^^^MD^^^UAMC^L|MED|||||A0||13579^POTTER^SHERMAN^T^^^
MD^^^UAMC^L

OBX|1|NM|HT^HEIGHT^99LOC1||71|in^inches^ANSI+|||||F

OBX|2|NM|WT^WEIGHT^99LOC1||175|lb^pounds^ANSI+|||||F

AL1|1|DA|ASP^ASPIRIN^99LOC2|MO|GI DISTRESS
```

CitiusTech

Message Framework (3/9)

HL7 Message structure version 2



Cardinalities

- [ ] = 0..1
- { } = 1.. *
- [{ }] = 0..*
- Else 1..1

Message Framework (4/9)

Message

- A message is the atomic unit of data transferred between systems
- 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 Patient Administration (ADT) data from one system to another
- There is a one-to-many relationship between message types and trigger event codes
- A message type may be associated with more than one trigger event
- A message is comprised of a group of segments in a defined sequence

Message Framework (5/9)

Segments and segment groups

- 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.
- Each segment is identified by a unique three-character code known as the Segment ID.
- All segment ID codes beginning with the letter Z are reserved for locally defined segments.
- Two or more segments may be organized as a logical unit called a segment group

Fields

- A field is a string of characters. Fields for use within HL7 segments are defined by HL7.

Message Framework (6/9)

Example Patient Registration Message

```
MSH|^~\&#|MegaReg|UABHospC|ImgOrdMgr|UABImgCtr|20010529090131-
0500||ADT^A01|01052901|P|2.7

EVN|A01|200105290901||||200105290900

PID|||56782445^^^UAReg^PI~999855750^^^USSSA^SS||KLEINSAMPLE^BARRY^Q^JR#||19
620910|M||C|260 GOODWIN CREST
DRIVE^^BIRMINGHAM^AL^35209^^H|||||||0105I30001

PV1||I|W^389^1^UABH^^^^3||||12345^MORGAN^REX^J^^^MD^^^UAMC^L||67890^GR
AINGER^LUCY^X^^^MD^^^UAMC^L|MED|||||A0||13579^POTTER^SHERMAN^T^^^MD^^
^UAMC^L

OBX|1|NM|HT^HEIGHT^99LOC1||71|in^inches^ANSI+|||||F

OBX|2|NM|WT^WEIGHT^99LOC1||175|lb^pounds^ANSI+|||||F

AL1|1|DA|ASP^ASPIRIN^99LOC2|MO|GI DISTRESS
```

CitiusTech

Message Framework (7/9)

MSH – Message Header

SEQ	LEN	C.LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	1..1		ST	R			00001	Field Separator
2	4..5		ST	R			00002	Encoding Characters
3			HD	O		0361	00003	Sending Application
4			HD	O		0362	00004	Sending Facility
5			HD	O		0361	00005	Receiving Application
6			HD	O		0362	00006	Receiving Facility
7			DTM	R			00007	Date/Time of Message
8		40=	ST	O			00008	Security
9			MSG	R			00009	Message Type
10	1..199	=	ST	R			00010	Message Control ID
11			PT	R			00011	Processing ID
12			VID	R			00012	Version ID
13			NM	O			00013	Sequence Number

Message Framework (8/9)

Message Header

MSH|^~\&#|MegaReg|UABHospC|ImgOrdMgr|UABImgCtr|200105290 90131-0500||ADT^A01|01052901|P|2.7

MSH	: Segment ID
|	: Field separator, must be printable, fixed length
^~\& #	: Encoding characters (component, repetition, escape, subcomponent separator, truncation character)
MegaReg	: Sending Application
UABHospC	: Sending Facility
ImgOrdMgr	: Receiving Application
UABImgCtr	: Receiving Facility
20010529091310	: Date and Time
ADT^A01	: Message Type
01052901	: Message Control ID
P	: Processing Type
2.7	: Version Id

CitiusTech

Message Framework (9/9)

- Message type **:** Indication that the data in the HL7 message belongs to a particular set of actions.
    - ADT : Admission, Discharge, Transfer
    - ACK : Acknowledgment

- Triggers : Creates the need for data to flow between healthcare systems.
    - A01 : Inpatient Admission
    - A04 : Outpatient Registration
    - A05 : Pre-Admission
    - A11 : Cancel Admission/Registration

Agenda

- Introduction
- Conceptual Approach
- Communications Environment
- Message Framework
- **Message Construction Rules**
- Use of Escape Sequences
- Version Compatibility Definition
- Acknowledgment Messages
- Message Control Segments

Message Construction Rules (1/15)

Message Delimiters

Delimiter	Suggested Value	Encoding Character Position	Usage
Segment Terminator	<cr>	-	Terminates a segment record. This value cannot be changed by implementers.
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.
Subcomponent 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 use with any field represented by an ST, TX or FT data type, or for use with the data (fourth) component of the ED data type. 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.

Message Construction Rules (2/15)

Length

- Normative Length

  - For some fields or components, the value domain of the content leads to clearly established boundaries for minimum and/or maximum length of the content.
  - In these cases, these known limits are specified for the item.
  - Normative lengths are only specified for primitive data types.
  - Examples of Value Domain are
    - Date/Time
    - A component that may contain values are ABC, SYL etc
    - A component that contains reference to a field in a message
  - The information is given in one of two forms:
    - the minimum and the maximum length separated by two dots, e.g. m..n
    - the list of possible values for length separated by commas, e.g. x,y,z

- The minimum length is always 1 or more.
- If an item is optional, and there is no content present, the item is considered as not populated, rather than present with a length of 0.

Message Construction Rules (3/15)

Length

- Length & Persistent Data Stores

  - For many fields or components, the value domain of the content does not lead to clearly established boundaries for minimum and/or maximum length of the content.
  - Examples of value domains are
    - Parts of Names and Addresses
    - Codes defined in external code systems
    - Descriptive text

  - In many cases, systems store the information of these value domains using data storage mechanisms that have fixed lengths, such as relational databases, and must impose a limitation on the amount of information that may be stored.
  - Though this does not directly impact on the length of the item in the instance, nevertheless the storage length has great significance for establishing interoperability.

Message Construction Rules (4/15)

Truncate Pattern

- Many applications may define a limit to the length that they will store for a particular item.
- So when the item exceeds the length limit set by the application there are 3 things that can be done
  - Data must be rejected
  - Truncate the data

- Data should be rejected for some data items which has clinical information but data items which represent name/address information can still be functional with truncation of data.

- Sometimes truncation of data may lead to problems – For example -
  - The surname is truncated and send back to the source application
  - The source application needs to search the patient based on the surname
  - The source application will fail since the surname was truncated.

Message Construction Rules (5/15)

Truncate Pattern

- The rule to be followed while truncation is to truncate the value at N-1
- Where N is length limit and final character replaced with a truncation character
- This means that whenever that value is subsequently processed later, either by the system, a different system, or a human user, the fact that the value has been truncated has been preserved, and the information can be handled accordingly.

- The truncation character is not fixed; applications may use any character.
- The truncation character used in the message is defined in MSH-2.
- The default truncation character in a message is # (23)
- The truncation character only represents truncation when it appears as the last character of a truncatable field.
- It may be escaped but this is not required.

Message Construction Rules (9/15)

Abstract Message Syntax

- Schema of the message?
- A definition table that indicates the usage of all the segments that may appear in the message legally in a standard HL7 message.
- Reading the syntax :
  - [] → 0..1
  - {} → 1..*
- Example
  - MSH MSA [ERR]
  - MSH PID PV1 {[ORC]{OBX}}

Message Construction Rules (10/15)

Given the following abstract message definition:

```
MSH Message Header
MSA Message Acknowledgment
[ { PID Patient Identification
    [ { WDN Widget Description
        { WPN Widget Portion }
        [ WPD Widget Portion Detail ]
      }
    ]
  }
]
```

Which of the following ordered segments in a message would be ILLEGAL?

A. MSH MSA PID
B. MSH MSA PID WDN WPN WPD PID PID PID
C. MSH MSA PID WDN WPN WDN WPN WPN WPD
D. MSH MSA PID WDN WPN WPD WPD WDN WPN
E. MSH MSA PID WDN WPN WDN WPN WPD

CitiusTech

Message Construction Rules (11/15)

Message construction rules

- Message consists of multiple segments.
- Segment consist of :
  - Segment ID (e.g. MSH, MSA, PID ..) first 3 characters
  - Field separator
  - Data fields

“End each segment with an ASCII carriage return character”

Message Construction Rules (12/15)

Insertion of fields in segment

- If the value is not present, no further characters are required
- Null values are represent as “”
- It is not necessary, and is undesirable, to pad fields to fixed lengths. Padding to fixed lengths is permitted.
- If field is having more than one component then components are separated with component separators.
- For more than one occurrence of field, repetition separator is used.
- For e.g. two occurrences of telephone number are being sent as:
  |(234)567-7120~(599)128-1234|

Message Construction Rules (13/15)

Component in a field

- If more than one component is included they are separated by the component separator
- Components that are present but null are represented by the characters ""
- Components that are not present are treated by including no characters in the component
- 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|
- Certain component definition calls for components to be broken into sub component

Similar rules apply for subcomponents

Standard subcomponent separator is “&”

Message Construction Rules (14/15)

Rules for receiving HL7 messages and converting their contents to data values

- Ignore segments, fields, components, subcomponents, and extra repetitions of a field that are present but were not expected.
- Treat segments that were expected but are not present as consisting entirely of fields that are not present.
- Treat fields and components that are expected but were not included in a segment as not present.

Message Construction Rules (15/15)

Messages may be locally extended as follows

- Users may develop local Z messages to cover areas not already covered by existing HL7 messages. These should be composed of HL7 segments where possible.
- A local Z message may consist entirely of Z segments except that it must begin with a MSH segment.
- A local Z Acknowledgement message must begin with an MSH segment followed by an MSA segment, an optional SFT segment and a conditional ERR segment.
- Users may develop Z segments and add them to Z messages.

Agenda

Use of Escape Sequences (1/2)

Use an escape sequence in place of the character to be sent

\F\ sends the field separator

\S\ sends the component separator

\T\ sends the subcomponent separator

\R\ sends the repetition separator

\E\ sends the escape character

\H\ start highlighting

\N\ normal text (end highlighting)

\L\ sends the truncation character

Use of Escape Sequences (2/2)

.ti <number> = Temporarily indent <number> spaces (at the beginning of a line only).

```
|\.ti 5\The patient breathes with difficulty.\.br\\.ti 5\There is history
of respiratory illness on both sides of the patient's family.|
```

would display as

```
          The patient breathes with difficulty.
          There is history of respiratory illness on both sides of the patient's
family.
```

Agenda

- Introduction
- Conceptual Approach
- Communications Environment
- Message Framework
- Message Construction Rules
- Use of Escape Sequences
- **Version Compatibility Definition**
- Acknowledgment Messages
- Message Control Segments

Version Compatibility Definition (1/2)

- The encoding rules (for receiving HL7 messages and converting their contents to data values) allow the following definition of a backward compatibility requirement between the 2.x versions of HL7:
- The HL7 Standard may introduce new messages in each succeeding version.
- The Standard may introduce new segments to an existing message in each succeeding version. In general, new segments will be introduced at the end of a message, but they may be introduced elsewhere within the message if the segment hierarchy makes this necessary.
- In each succeeding version, the Standard may add new fields at the end of a segment, new components at the end of a field, and new subcomponents at the end of a component; and a non-repeating field may be made repeating.

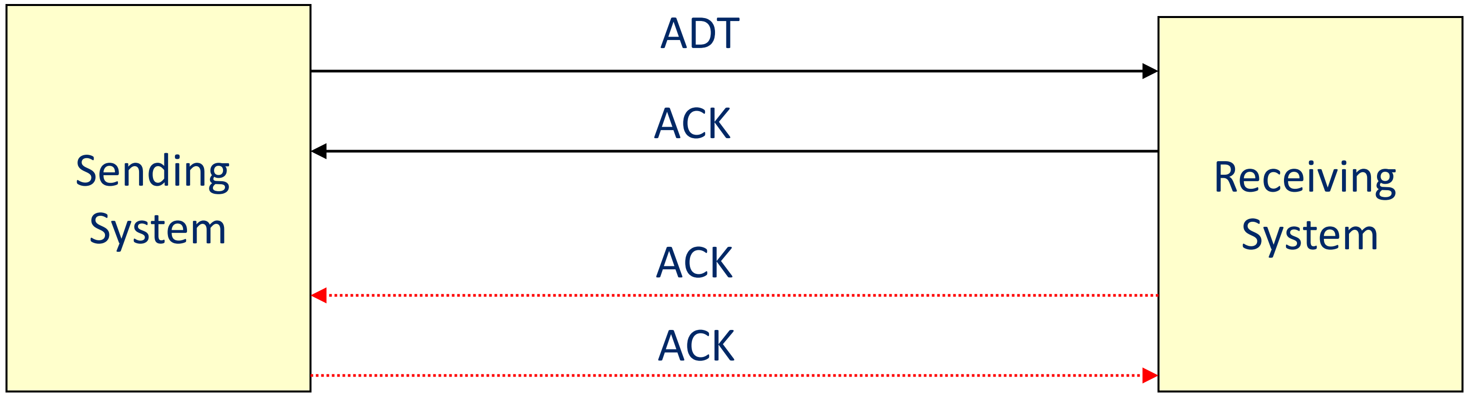
Version Compatibility Definition (2/2)

- If a non-repeating field is made repeating, the first instance of that repeating field must have the same meaning as the non-repeating field had in the prior version of HL7.
- For existing fields in existing segments, data types may be changed by the rule mentioned above, if the leftmost (prior version) part of the field has the same meaning as it had in the prior version of HL7.
- In other words, if the new parts of the field (those that are part of the new data type) are ignored, what remains is the old field (defined by the old data type), which has the same meaning as it had in the prior version of HL7.

Agenda

Acknowledgement Messages (1/6)

Message/Acknowledgement protocol



- Receiver informs the sender whether the message was received and processed successfully.
- If a problem, can isolate details and location within message.
- Original Mode ACK : point to point setting
- Enhanced Mode ACK: network settings

Acknowledgement Messages (1/6)

```
MSH|^~\&|ImgOrdMgr|UABImgCtr|MegaReg|UABHospC|20070529090142-
0500||ACK^A01 |3944441|P|2.5
MSA|AA|01052901
```

```
MSA|AA|01052901
```

The MSA segment contains acknowledgement information such as

– The acknowledgement code (indicating success or failure)

– The message control ID of the message being acknowledged

– Optional error segment

Fields of interest in the MSA segment:

Sequence	Name
1	Acknowledgement Code
2	Message Control ID
3	Text Message
6	Error Condition

Acknowledgement Messages (3/6)

- HL7 defines acknowledgement rules that allow the receiving system and/or application to give the following information to the sending system and/or application:

  Whether the message was received properly.

  Whether the message was processed properly.

- **Original Mode ACK : point to point setting**

  When the unsolicited update is sent from one system to another, this acknowledgement mode specifies that it be acknowledged at the application level.

- **Enhanced Mode ACK: network settings**
  1. the receiving system may send back an immediate accept acknowledgement indicating whether it was able to receive and take custody of the message, without respect to the status of further processing.
  2. Later on, the receiving application may send back an application acknowledgement indicating whether the message could be processed.

Acknowledgement Messages (4/6)

The following is the processing flow for original mode acknowledgement:

- If the originating message has an invalid value for *message type (MSH9), version ID (MSH12), or processing ID (MSH11), the receiver returns an acknowledgement* message with an acknowledgement code of AR (application reject), and processing is complete.
- If the originating message is not processed successfully, the receiver returns an acknowledgement message with an acknowledgement code as follows:
  - AE (application error) if the content or format of the message were invalid (e.g., missing segments, invalid order code). In this also be sent. case an ERR segment may
  - AR if the message failed processing for some other reason (e.g., system down, queue full).
- Otherwise, if the originating message is processed successfully, the receiver returns an acknowledgement message with an acknowledgement code of AA (application accept), and processing is complete.

Acknowledgement Messages (5/6)

The following is the processing flow for enhanced mode acknowledgement:

- If the originating message has an invalid value for message type (MSH9), version ID (MSH12), or processing ID (MSH11), the receiver returns an acknowledgement message with an acknowledgement code of CR (Commit reject), and processing is complete.

- Otherwise, if the originating message cannot be accepted for some other reason (e.g., the value of sequence number is invalid), the receiver returns an accept acknowledgement message with an acknowledgement code of CE (commit error) and a further description of the error in text message.

- Otherwise, the originating message is accepted successfully, the receiver returns an acknowledgement message with an acknowledgement code of CA (commit accept), and processing is complete.

Acknowledgement Messages (6/6)

Points to remember:

1. For original mode: Both MSH-15-accept acknowledgement type and MSH-16-application acknowledgement type are null or not present.

2. For enhanced mode: At least one of MSH-15-accept acknowledgement type or MSH-16-application acknowledgement type is not null.

3. 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

Note: AL – Always, NE – Never

Agenda

Message Control Segment (1/2)

Message Control Segments

Segment Category	Segment Name	Description	Purpose
Control	ADD	Addendum segment	Used to define the continuation of the prior segment
	BHS	Batch header segment	Defines the start of a batch
	BTS	Batch trailer segment	Defines the end of a batch
	DSC	Continuation pointer segment	Used in the continuation protocol
	ERR	Error segment	Used to add error comments to acknowledge messages
	FHS	File header segment	Used to head a file (group of batches)
	FTS	File trailer segment	Defines the end of file
	MSA	Message acknowledgment segment	Contains information sent while acknowledging another message
	MSH	Message header segment	Defines the intent, source, destination, and some specifics of the syntax of a message

Message Control Segment (2/2)

Segment Category	Segment Name	Description	Purpose
General Purpose	NTE	Notes and comments segment	Used for sending notes and comments
	OVR	Override segment	This segment allow a sender to override specific receiving application’s business rules to allow for processing of a message that would normally be rejected or ignored
	SFT	Software segment	This segment provides additional information about the software product(s) used as a Sending Application. The primary purpose of this segment is for diagnostic use.

Thank You