Chapter 6
HL7 Version 2

HL7 Version 2 is the most widely used healthcare interoperability standard in the world. It is used in over 90% of all hospitals in the USA and is widely supported by healthcare IT suppliers worldwide.

At first sight, the HL7 V2 documentation may appear to be large and formidable, but it is based on a few basic principles, which are quite easy to grasp.

To understand some of the features of HL7, we need to go back to its origins in 1987. The initial focus of HL7 was on exchanging information about admissions, discharges, and transfers (ADT) within hospitals. The first version, HL7 V1.0 was issued a few months later. In the following year, 1988, HL7 V2.0 was published, and this included a major extension to add in messages for exchanging orders and reports for tests and treatment, based closely on the ASTM (American Society of Testing and Materials) E.1238.88 standard. Version 2.1, which was the first widely used version, was published in 1991.

The HL7 V2 standard has been in continuous development for over 20 years. At the time of writing, the latest version is Version 2.6, which was approved as an ANSI standard in October 2007. Version 2.6 is organized as a set of chapters and appendices as shown in Fig. 6.1:

During its long development period the scope of HL7 Version 2 has increased enormously, but the basic principles have hardly changed. The Version 2.6 standard now has 1,965 pages and 717,000 words. It contains an enormous amount of knowledge and experience about health informatics.

One of the basic principles of HL7 V2 has been the preservation of backward compatibility, while the standard has evolved by addition. The idea being that a system, which can understand a new message in a new version, should also be able to understand a previous version. Ideas, which have been superseded, are flagged as being deprecated, but not replaced.

Older versions are still widely used, because there is minimal return on investment achieved by replacing a working interface with a later version and a significant risk of hitting unexpected problems. However, interface engineers may need to work with several different versions and recognize the differences between them. It is always important to know what version is being used.

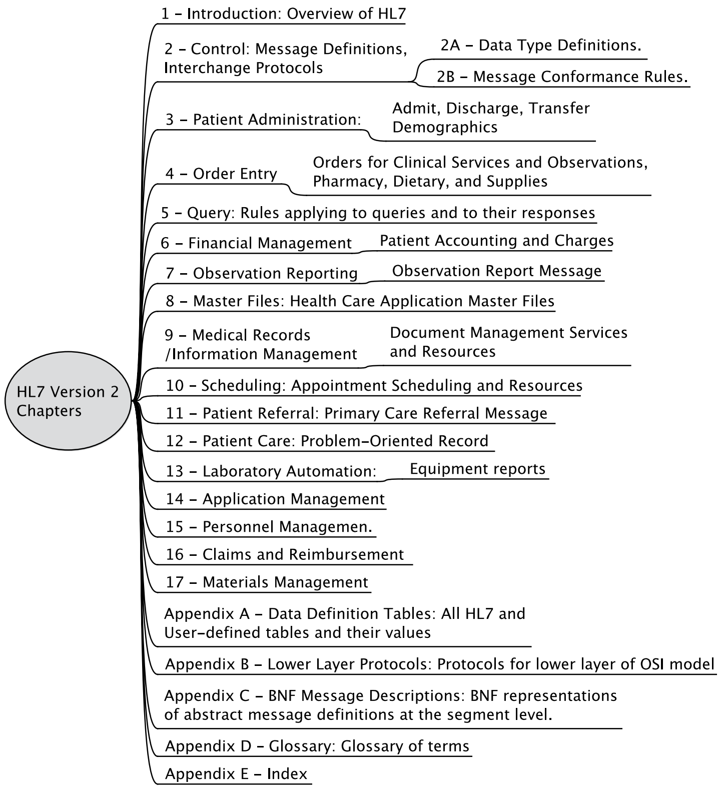
HL7 Version 2 Chapters
1 – Introduction: Overview of HL7
2 – Control: Message Definitions, Interchange Protocols
2A – Data Type Definitions.
2B – Message Conformance Rules.
3 – Patient Administration:
Admit, Discharge, Transfer Demographics
4 – Order Entry
Orders for Clinical Services and Observations, Pharmacy, Dietary, and Supplies
5 – Query: Rules applying to queries and to their responses
6 – Financial Management
Patient Accounting and Charges
7 – Observation Reporting
Observation Report Message
8 – Master Files: Health Care Application Master Files
9 – Medical Records /Information Management
Document Management Services and Resources
10 – Scheduling: Appointment Scheduling and Resources
11 – Patient Referral: Primary Care Referral Message
12 – Patient Care: Problem–Oriented Record
13 – Laboratory Automation:
Equipment reports
14 – Application Management
15 – Personnel Managemen.
16 – Claims and Reimbursement
17 – Materials Management
Appendix A – Data Definition Tables: All HL7 and User–defined tables and their values
Appendix B – Lower Layer Protocols: Protocols for lower layer of OSI model
Appendix C – BNF Message Descriptions: BNF representations of abstract message definitions at the segment level.
Appendix D – Glossary: Glossary of terms
Appendix E – Index

Fig. 6.1 HL7 Version 2 chapters

To understand the HL7 V2 documentation, you need to know about the message syntax and data types.

Message syntax describes the overall structure of messages and how the different parts are recognized. Each message is composed of segments in specified sequence, each of which contains fields also in a specified sequence; these fields have specified data types.

Data types are the building blocks of the fields and may be simple, with a single value, or complex, with multiple components. These components themselves have data types, which can be simple or complex, leading to subcomponents (Fig. 6.2).

6.1 Message Syntax

HL7 V2 messages are sent in response to trigger events. The message name is derived from the message type and a trigger event. The message type is the general category into which a message fits. For example, patient administration

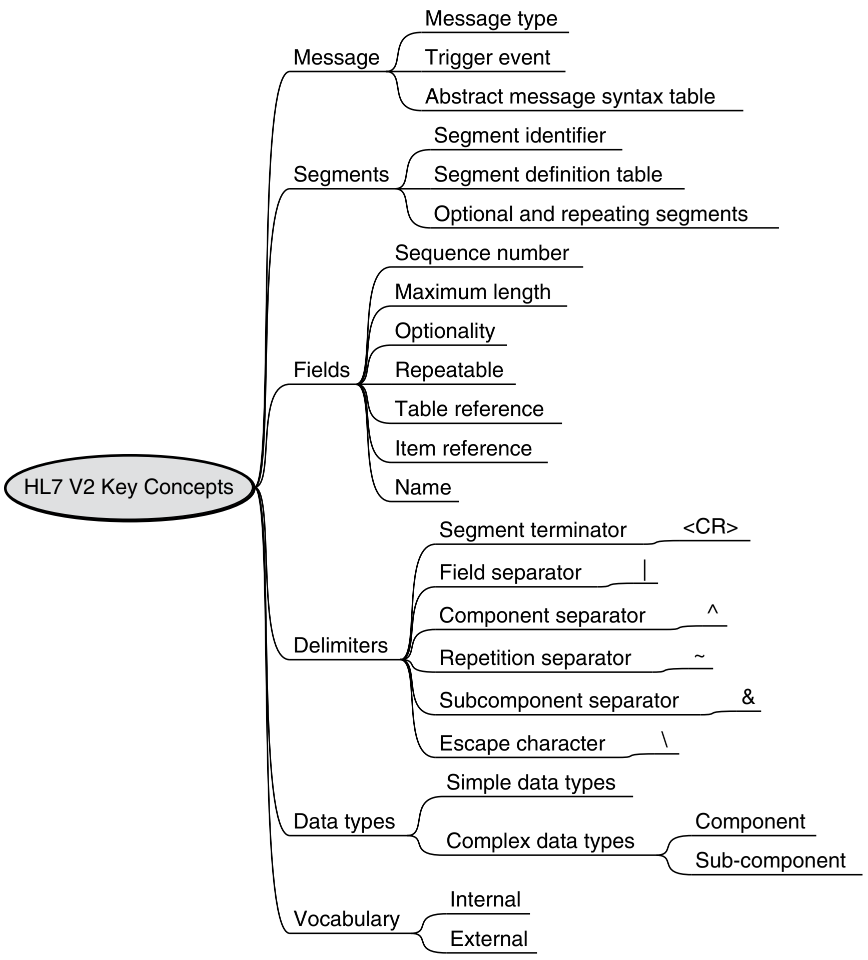

Fig. 6.2 HL7 Version 2 key concepts

messages are ADT. Examples of message types are shown in the following table, which also shows the HL7 V2 Chapter where they are described in detail:

Value	Description	V2 Chapter
ACK	General acknowledgment message	2
ADT	ADT message	3
ORM	Order message	4
ORU	Observation result – unsolicited	7

The trigger event indicates what happened to cause a message to be generated. Trigger events are specific to a message type. For example some of the ADT trigger events are:

Value	Description
A01	Admit/visit notification
A02	Transfer a patient
A03	Discharge/end visit
A04	Register a patient

The full message name is ADT^A01 (the "^" is the HL7 field component separator). The message name is always entered in the ninth field of the message header segment (MSH-9).

Each HL7 V2 message comprises a set of segments. The overall structure and allowable content of each message is defined in an abstract message syntax table, which lists segments in the order in which they occur. For example, a simple message, noting that a patient has been admitted to the hospital contains the following segments in the order shown:

MSH Message Header
EVN Event Type
PID Patient Identification
PV1 Patient Visit

The abstract message syntax also shows which segments are optional and which can be repeated. A segment listed on its own is mandatory and may not repeat. Optional segments are surrounded by square brackets [. . .]. Segments that are allowed to repeat are indicated using curly braces { . . . }. If a segment is both optional and repeatable, it has both brackets and braces [{ . . . }]. Note that the order is not important: [{...}] and {[...]} are equivalent.

The abstract message syntax table also shows which chapter of the HL7 V2.x standard contains the segment definition. Part of the abstract message syntax table for the ADT^A01 message is shown below:

ADT^A01	ADT Message	Chapter
MSH	Message Header	2
EVN	Event Type	3
PID	Patient Identification	3
[PD1]	Additional Demographics	3
[{ NK1 }]	Next of Kin/Associated Parties	3
PV1	Patient Visit	3

This shows that segments MSH, EVN, PID, and PV1 are mandatory. PD1 is optional. It is also indented, which indicates that it is nested under the PID segment, creating a group. NK1 is both optional and repeatable.

Each segment has a three-character identifier, the segment ID (e.g., MSH). In a message this segment ID is always the first three characters of the line.

Segments contain fields and fields contain components; components may contain subcomponents.

6.1.1 Delimiters

Delimiters (such as field separators, component separators, and subcomponent separators) are used to indicate the boundaries between these elements. The term element is used to refer to a field, a component, or a subcomponent.

Within the HL7 syntax, the size of messages transmitted is reduced by truncation. If fields at the end of a segment or component are not needed, the appropriate terminator or separator character truncates them. The segment terminator (carriage return) truncates segments. In the same way, field separators truncate components and component separators truncate subcomponents.

Most HL7 V2 implementations use default encoding with the delimiters to terminate segments and to separate components and subcomponents. The delimiters are defined in the first two fields of the MSH segment (MSH-1 and MSH-2). There is also an XML representation (not described here).

Symbol	Usage
|	Field separator
^	Component separator
~	Repetition separator
\	Escape character
&	Subcomponent separator
<CR>	Segment terminator

The field separator (|) is always the fourth character of each segment. Fields are named according to their sequential position within a segment. For example, MSH-9 is the ninth field in the MSH segment and is preceded by nine field delimiters. Two adjacent field separators (||) indicate an empty field. If an application wishes to state that a field contains null and expects the receiving system to act on this, then an explicit null is represented as |””|.

The component separator (^) separates the components of a field. Components are referred to by the segment, field, and position in the field (e.g., MSH-9.1). For example, the MSH-9 field contains two components: MSH-9.1 (message type) and MSH-9.2 (trigger event) and might be represented as ADT^A01. The field separator truncates any components, not needed at the end of a field. For example, the following two data fields are equivalent: |ABC^DEF^^| and |ABC^DEF|.

The repetition separator (~) is used to separate the first occurrence or repetition of a field from the second occurrence and so on.

The escape character (\) is used mainly in text elements to bracket text for special processing. The escape character can be used to send delimiters within a message.

Symbol	Escape sequence
|	\F\
^	\S\
~	\R\
\	\E\
&	\T\ e.g., |Marks \T\ Spencer|

The escape character may also be used to indicate certain formatting commands, such as \.br\ to indicate line break, or \.sp 3\ to skip 3 spaces in the formatted text (FX) data type.

The subcomponent separator (&) is used to separate subcomponents within components, providing an additional level of granularity.

Each segment is ended with an ASCII carriage return<CR>character.

6.2 Segment Definition

Each segment is defined in a table such as that shown below for the MSH Message Header segment. All HL7 V2 messages begin with a single MSH segment and this provides an example of how segments are defined.

SEQ	LEN	DT	OPT	RP/#	TBL#	ITEM #	ELEMENT NAME
1	1	ST	R			00001	Field Separator
2	4	ST	R			00002	Encoding Characters
3	180	HD	O			00003	Sending Application
4	180	HD	O			00004	Sending Facility
5	180	HD	O			00005	Receiving Application
6	180	HD	O			00006	Receiving Facility
7	26	TS	O			00007	Date/Time Of Message
8	40	ST	O			00008	Security
9	7	CM	R			00009	Message Type
10	20	ST	R			00010	Message Control ID
11	3	PT	R			00011	Processing ID
12	8	ID	R		0104	00012	Version ID
13	15	NM	O			00013	Sequence Number
14	180	ST	O			00014	Continuation Pointer
15	2	ID	O		0155	00015	Accept Acknowledgment Type
16	2	ID	O		0155	00016	Application Ack Type
17	2	ID	O			00017	Country Code
18	6	ID	O	Y/3	0211	00692	Character Set
19	60	CE	O			00693	Principal Language Of Message

The columns of this table show:

SEQ: Field sequence number
LEN: Maximum field length
DT: Data type
OPT: Optionality: R (required), O (optional), C (conditional), B (deprecated but retained for backward compatibility)

RP/#: Repeatable field. If "y" can repeat any number of times; a number indicates the maximum number of repeats
TBL#: The reference number of the HL7 table which contains a controlled vocabulary from which values can be taken
ITEM#: HL7's internal database item number
ELEMENT NAME: Human readable name of the field

Other required fields in addition to the Field Separator and Encoding Characters are:

- HL7V2 Message Type is entered in MSH-9.1, which is the first component of field MSH-9.
- Trigger Event is entered in MSH-9.2, which is the second component of field MSH-9.

We now describe some of the commonly used segments (Fig. 6.3). It might be an idea to bring Fig. 6.3 forward.

6.2.1 Message Header MSH

The report header (MSH) contains common metadata found in most messages, irrespective of subject. The first two fields of the MSH segment specify the delimiters used (see above).

SenderID is a unique identifier for the sender, expressed as the combination of an identification code for the sender and a code for the naming authority that controls the assignment of these identification codes. The only constraint is that the combination of MSH-4.2 and MSH-4.3 is unique.

For example: |^123457^Labs|.

DateTime of message is the exact date/time, that the sending system created the message. It is held in field MSH-7. For example |20080805183015+0000| indicates Aug 5, 2008 6.30 pm and 15 s GMT.

MessageType is the HL7 V2 message type and trigger event, transmitted in field MSH-9. Laboratory report messages all have the content |ORU^R01|.

MessageID is used to uniquely identify the message. This is transmitted in field MSH-10. The sending system must assign an identifier, which is unique to the extent in combination with the SenderID it is globally unique. One way of ensuring uniqueness is to use a globally unique identifier such as a GUID, which is produced on the fly by software. However, GUIDs are longer than the 20 characters prescribed by HL7.

ProcessingStatus shows whether the message is production (P) or for some other use such as debugging (D) or training (T). Production messages have the code P in field MSH-11, |P|.

SyntaxVersion indicates the HL7 version with which this message claims compliance. Compliance with HL7 V2.4 is shown by entering 2.4 in field MSH-12, |2.4|.

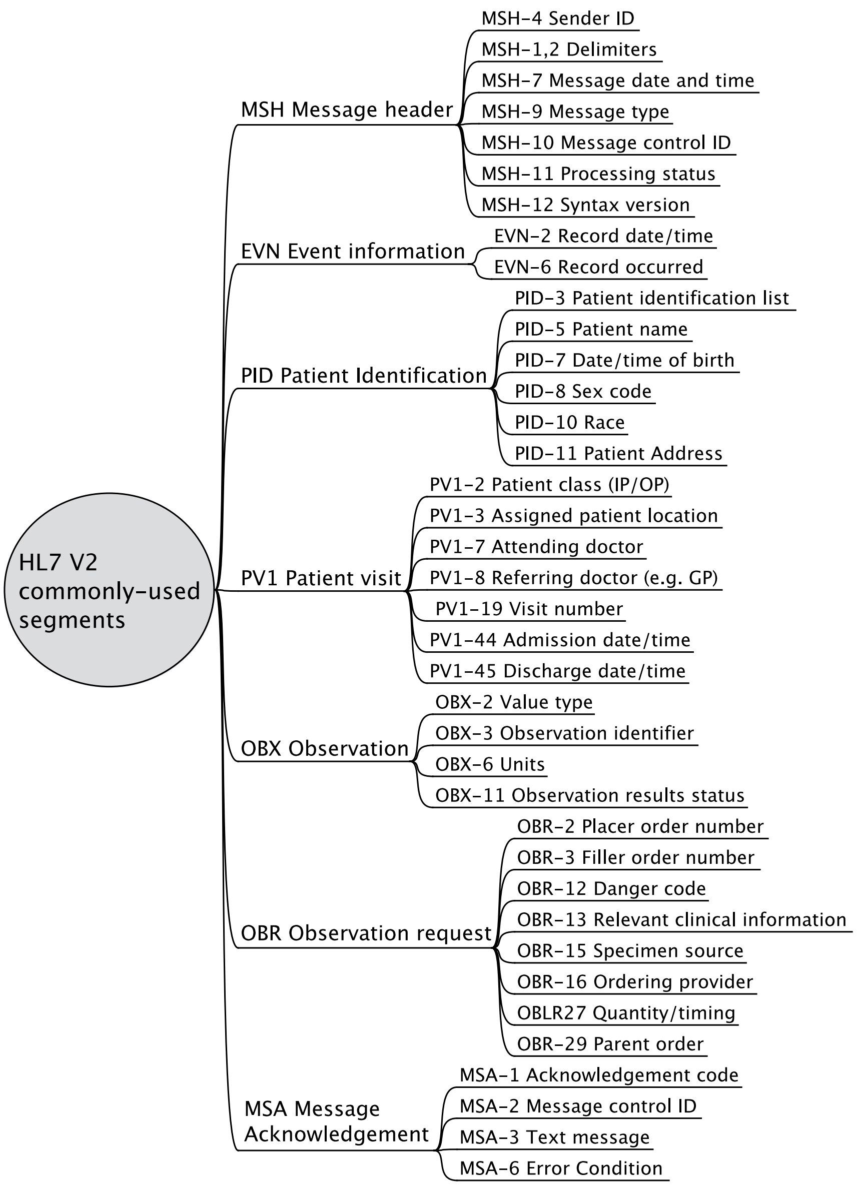

Fig. 6.3 HL7 V2 commonly used segments

6.2.2 *Patient Identification Details (PID)*

PatientID refers to the patient identifiers (one or more), which are used by the healthcare facility to uniquely identify a patient (e.g., hospital number, NHS number). In HL7 V2 these identifiers are sent in field PID-3, with the identifier in the first component (PID-3.1), an optional identifier for the issuing authority in the fourth component (PID-3.4), and an identifier type code (required) in the fifth component (PID-3.5).

For example, a patient with hospital number 123456 at St Mary's Hospital (SMH) may be entered as |123456^^^SMH^PI|, where PI indicates that this is a Patient internal identifier.

If the sender only uses the NHS number, e.g., 9999999904, this could be exchanged as |9999999904^^^NHS^NH|.

The repetition separator, ~, separates the combination of both hospital number and NHS number, together:

|123456^^^SMH^PI ~ 9999999904^^^NHS^NH|

PatientName includes the first (given) and last (family) name of the patient. These are provided in fields PID-5.1 and PID-5.2 respectively.

Mary Smith would become |Smith^Mary|.

DateOfBirth is recorded as a date in field PID-7 in format YYYYMMDD.

e.g., |19620114| for 14 January 1962.

SexCode is in field PID-8, using an agreed coding system, such as M = Male and F = Female, e.g., |M|.

Patient address is transmitted in field PID-11, using the following components:

Street address	PID-11.1
Second line of address	PID-11.2
City	PID-11.3
State, province or county	PID-11.4
Zip or postal code	PID-11.5
Country	PID-11.6

For example |14 Pinewood Crescent^Hermitage^^^RG18 9WL| shows two lines of address and a postcode.

6.2.3 Patent Visit (PV1)

The PV1 (patient visit) segment is used in this example for both the patient's GP and the patient location at which the sample was taken.

Patient Location is the location at which the sample was taken. This information is mandatory for infection control. It is exchanged using the PV1-3 field (assigned patient location), using a mutually agreed code.

General Practitioner (person responsible for the Patients Health in the Community). The patient's GP identifier is shown in the PV1 segment, field PV1-8. In most cases only an agreed identifier is sent in component PV1-8.1. This data is desirable but not mandatory.

6.2.4 Request and Specimen Details (OBR)

The laboratory allocates each specimen an accession number, which is used to identify that specimen and any derivatives. In HL7 this is referred to as the Filler Order Number and is provided in field OBR-3.1.

Lab Test Code records what was requested to be done. An agreed code system, such as LOINC should be used. It is provided in field OBR-4, component OBR-4.1 with the text name in component OBR-4.2 and the name of coding system in OBR-4.3.

The date and time that the specimen was collected from the patient is provided in field OBR-7, using format YYYYMMDDHHMM. The time is optional.

The specimen source is provided using an agreed code or controlled vocabulary in field OBR-15.1 (e.g., WOUND SWAB).

Body Site (desirable) states the part of the body from which the specimen is taken. This is provided as a string in field OBR-15.4 (e.g., FOOT).

Site Modifier (optional) is sometimes reported, to provide additional information about the body site. If used it is provided in field OBR-15.5 (e.g., Right).

The doctor who ordered the test is recorded in field OBR-16, using an agreed identifier in OBR-16.1.

6.2.5 Result Details (OBX)

Each separate result is entered as a separate OBX segment, which relates to a single observation or observation fragment. It represents the smallest indivisible unit of a report.

Each result is represented as an attribute–value pair.

The *data type* of the value is specified in OBX-2. In HL7 terminology the attribute being measured is specified in OBX-3 (Observation Identifier) and the value is in OBX-5 (Observation Value). Internal references are specified in OBX-4.

In Microbiology, organisms, or the presence of an organism, are identified by either isolating the organism on a medium, or testing for the presence of an organism using a variety of tests. Isolates generally have associated antibiotic susceptibilities.

Observation Identifier (OBX-3) is the test that is being done (the attribute being measured) and typically uses LOINC or locally defined codes. Field OBX-3.1 contains the code; OBX-3.2 contains the human-readable display text; OBX-3.3 contains the coding scheme identifier if used.

For example: |9999-9^Test name^LN|.

(OBX-5) is the value of the result and typically uses SNOMED or SNOMED CT coding system.

The value type – the data type of the observation value – is specified in the Value Type (OBX-2).

The code value is OBX-5.1, display text is OBX-5.2, and code system identifier is OBX-5.3. Text strings can be transmitted in OBX-5.2 (e.g., |^This is a result|). Numeric values are represented as strings to allow non-numeric characters to be used (such as ">").

OBX|1|CE|5182-1^Hepatitis A Virus IgM Serum Antibody EIA^LN||G-A200^ Positive^SNM|

Some microbiology results have an extra complication. The first stage is to identify the various isolates (such as bacteria), which are present in the specimen. The second stage is to test each of these isolates for susceptibility to treatment by various antibiotics. The solution is to use internal references to link all of the results for the same isolate together using the Observation Sub-ID (OBX-4). Each OBX segment for the same isolate contains the same integer value in OBX-4.

The Abnormal Flag (OBX-8). If the observation is an antimicrobial susceptibility, the interpretation codes are: S = susceptible; *R* = resistant; I = intermediate; MS = moderately susceptible; VS = very susceptible.

The observation result status is required to indicate whether the result is Final, Preliminary, or otherwise and should be present in OBX-11.

6.2.6 Z-Segments

HL7 V2 provides a facility for any users to develop their own segments, message types, and trigger events using names beginning with Z. Z-segments are widely used and are one of the main reasons why there are so many different variants of HL7 V2 messages.

Z-segments can be placed anywhere in a message. Some message designers place all Z-segments at the end of a message, whilst others place them adjacent to related information.

6.3 Data Types

Data types are the basic building blocks used to construct or constrain the contents of each element. Every field, component, and subcomponent in HL7 V2 has a defined data type, which governs the information format in the element, what sub-elements it can contain and any vocabulary constraints. Some data types are Simple others are Complex.

HL7 V2 has 89 data types in all, but most applications use only a small number of common data types.

Simple data types contain just a single value, while complex data types may contain more than one sub-element, each of which has its own data type. The data type of a component can also be a complex data type. In this case, that component's components are subcomponents of the original data type. No further recursion is allowed.

Complex data types reflect associations of data that belong together, such as the parts of a person's name, address, or telephone number, or linking identifiers with their issuing authority (Fig. 6.4).

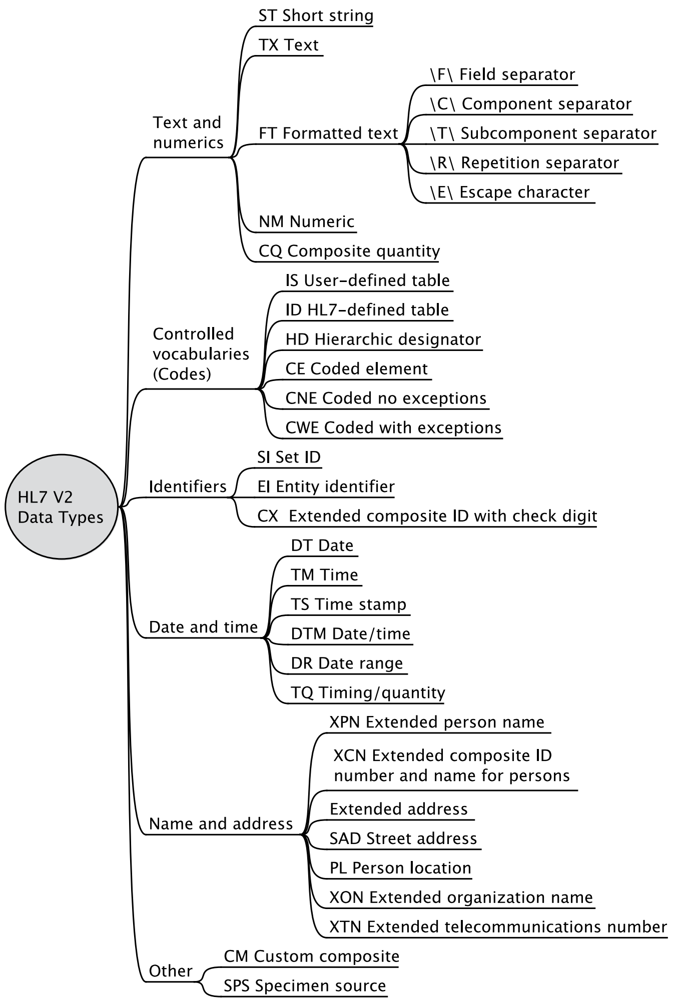

Fig. 6.4 HL7 V2 Data Types

6.3.1 Simple Data Types

Simple data types include:

- DT (date) represents a date in format: YYYY[MM[DD]]. For example, 2 August 2008 is represented as 20080802.
- DTM (date/time) is used to represent an event date and time including time zone if required. YYYY[MM[DD[HHMM[SS[.S[S[S[S]]]]]]]][+/–ZZZZ] where +/–ZZZZ indicates the time zone.
- FT (formatted text) allows embedded formatting commands, bracketed by the escape character.
- ID represents a value from a HL7-defined table. Users are not allowed to add their own values.
- IS represents a value from a user-defined table.
- NM (numeric) is used for numeric values. It may be preceded by a sign and may contain a decimal point.
- SI (set ID) gives the order of a segment instance within a message that may contain more than one segment with the same segment ID.
- ST (string) is used for short strings up to 200 characters.
- TX (text) is used for longer texts up to 64 K characters. In the TX data type the repetition separator (~) is used to indicate a hard carriage return (line break).

6.3.2 Complex Data Types

HL7 V2 supports a variety of complex data types to handle items such as coded values, identifiers, names, addresses, and so on. The most commonly used complex data types fall into three broad categories: Codes and Identifiers; Names and Addresses; and Other Complex Data Types.

6.3.3 Codes and Identifiers

Codes and identifiers are particularly important in interoperability and HL7 V2 supports both internally defined (by HL7) and externally defined coding schemes.

Coded values need to be uniquely identified, but there is always the problem that two different coding schemes use the same code value. The solution is to explicitly identify both the coding scheme and the code value. HL7 V2 enables this in two ways.

The first method is to populate the code values from a table, which is explicitly named in the data type definition. This method is used for the ID and IS data types. The second more general method is to transmit an agreed coding scheme identifier with each code value.

Coding schemes are either internal, which means defined by HL7, or external, which means defined by some other party. There are two main data types, CNE and CWE, which are very similar and replace the CE data type used in early versions of HL7 V2.

CNE (Coded with No Exceptions) is used when a required or mandatory coded field using a specified internal or external coding system must be used and may not be extended with local values. In CNE, the identifier component (CNE.1) is required.

CWE (Coded With Exceptions) is used when the set of allowable code values may vary on a site-specific basis or no code value is available for transmission, just a text string (CWE.2). CWE is very similar to CNE, but the identifier component (CWE.1) is optional.

The CNE and CWE data types have a similar structure:

Seq	Name	Data Type	Use
1	Identifier	ST	Code for the concept being represented
2	Text	ST	Name of the concept being represented
3	Name of Coding System	ID	Coding system identifier from which code value (in component 1) is selected
4	Alternate Identifier	ST	
5	Alternate Text	ST	
6	Name of Alternate Coding System	ID	Identifies coding system used for Alternate Identifier
7	Coding System Version ID	ST	
8	Alternate Coding System Version ID	ST	
9	Original Text	ST	Original text before coding

In most applications only the first three components of coded element are used. However, both CNE and CWE allow the expression of a single concept in two different coding schemes. This may be useful when a sending system holds data using a different coding system than that required by the destination. The original text may also be transmitted.

The commonly found complex data types for names and identifiers include:

- CE (coded element) can be used to represent an external code set or a non-coded text value.
- CNE (coded with no exceptions) strictly constrains an element to the values in a specified coding system.
- CWE (coded with exceptions) allows local codes or text to be used to supplement the specified coding system.
- CX (extended composite ID with check digit) is used for identifiers, including context and optional check digit information.
- EI (entity identifier) is used to specify identifiers.
- HD (hierarchic designator) is used to represent a code value or an identifier. It is useful for elements that some systems may treat as a code and other systems may treat as an identifier.

6.3.4 Names and Addresses

- FN (family name) surname.
- PL (patient location) within an institution; may include bed, room, ward, floor, building, facility, status, and type.
- SAD (street address), house number, and street.
- XAD (extended address) the full location address. Street address (SAD), city, state, postal code, country; also allows start and end dates.
- XCN (extended composite ID number and name for persons) is used for clinical staff. This is the largest data type with 23 components, combining the features of both CX and XPN into a single field.
- XON (extended organization name) is used for healthcare organizations.
- XPN (extended person name) is used for patients and their relatives. Includes family name (FN), given name(s), title, suffix, type, and date range.
- XTN (extended telecommunication number) is for electronic addresses including telephone and email. It includes optional codes for use (e.g., home or work) and type (e.g., direct line or mobile).

6.3.5 Other Complex Data Types

- CQ (composite quantity) has subcomponents quantity and units.
- SPS (specimen source) covers information about specimen type, body site, collection method, additives, etc.
- TQ (timing/quantity) allows the specification of the number, frequency, priority, etc. of a service, treatment, or test.

6.4 A Simple Example

The following example is from a simple feed of laboratory test reports from a microbiology laboratory to an infection control monitoring system. Each report includes:

- A header stating the type, origin, and date time of the message
- A single patient with ID number, name, sex, date of birth, address, and General Practitioner identifier
- Specimen details of the laboratory accession number (ID), source, body site, time of collection, and requester
- A set of test results, including the test name and result and abnormality flag

The abstract syntax of the HL7 V2 message is:

MSH Message header
PID Patient Identification Details
PV1 Patient Visit

OBR Results header
{OBX} Results detail (repeats)

All segments are required.

The structure of an HL7 V2 message which meets these requirements is:

```
MSH|delimiters||sender|||dateTime||messageType|messageID
  |processingStatus|syntaxVersion
PID|||patientID^^^source^IDtype||familyName^givenName||d
  ateOfBirth|sex|||streetAddress^addressLine2^^^postCode
PV1|||patientLocation||||patientsGP
OBR|||accessionNumber|testCode^testName^codeType|||speci
  menDate|||||||specimenSource^^^bodySite^siteModifier
  |requester
OBX||valueType|observableCode^observableName|observatio
  nSubID|valueCode^valueText^valueCodeType|||abnormalFl
  ag|||result status
OBX ...
```

A populated example is:

```
MSH|^~\&||^123457^Labs|||200808141530||ORU^R01|12345678
  9|P|2.4
PID|||123456^^^SMH^PI||MOUSE^MICKEY||19620114|
  M|||14 Disney Rd^Disneyland^^^MM1 9DL
PV1|||5N||||G123456^DR SMITH
OBR|||54321|666777^CULTURE^LN|||20080802|||||||SW^^^FO
  OT^RT|C987654
OBX||CE|0^ORG|01|STAU|||||F
OBX||CE|500152^AMP|01|||R|||F
OBX||CE|500155^SXT|01|||S|||F
OBX||CE|500162^CIP|01|||S|||F
```

This could be rendered as:

Report from Lab123457, 15:30 14-Aug-2008, Ref 123456789
Patient: MICKEY MOUSE, DoB: 14-Jan-1962, M
Address: 14 Disney Rd, Disneyland, MM1 9DL
Specimen: Swab, FOOT, Right, Requested By: C987654,
Location: 5N
Patients GP: Dr Smith (G123456)
Organism: STAU
Susceptibility: AMP R
SXT S
CIP S

Note that the OBX segment repeats. Information about the susceptibilities of organism detected (STAU – staphylococcus aureus) is linked to that organism finding by using the OBX-4 Observation Sub-ID field.