Chapter 16
HL7 Dynamic Model

Abstract The dynamic model specifies the interoperability message flow between two parties, including the interactions, trigger events that lead to them, application roles, message types, interaction sequence, and message wrappers. The implementation technology specification (ITS) describes the wire format.

Keywords Dynamic model • Interaction • Trigger event • Application role • Message type • Interaction sequence • Message wrapper • Acknowledgement • Implementation technology specification (ITS)

The following aspects of the Dynamic Model have to be specified (see Fig. 16.1):

- Interactions specify the message content the sender and receivers
- Trigger events determine when certain messages can be transmitted
- Application roles (sender and receiver) represent groupings of functionality that an application can do. This includes receiver responsibilities
- Message type(s)
- Interaction sequence
- Message wrappers
- Acknowledgements

Interaction

An interaction is the smallest unit (atomic) of communication that can stand on its own. It is a one-way transfer of information and ties together HL7's static models of payload content and the dynamic model of information flow and system behaviour.

Formally, an interaction is a unique association between a specific message type, a particular trigger event that initiates or triggers the transfer, and the application roles that send and receive the message type.

In HL7 Version 3, each interaction is described in a table with its name and artefact ID, together with the sending and receiving Application roles, the trigger event,

T. Benson, G. Grieve, *Principles of Health Interoperability*,
Health Information Technology Standards, DOI 10.1007/978-3-319-30370-3_16

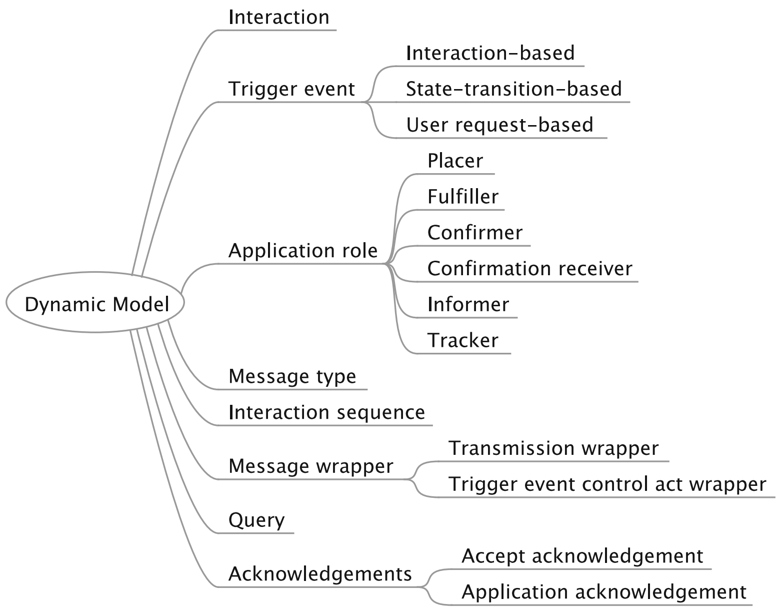

Fig. 16.1 HL7 dynamic model

the message type, the trigger event type and the wrapper types and their artefact identifiers.

Trigger Event

A trigger event is an explicit set of stated conditions, which can be recognized by a computer system, that initiates an interaction. A trigger event may be the result of human action, such as a mouse click, a state transition for an information object (such as the successful completion of a business transaction), an exception condition (such as an error), or be specific to a point in time (e.g. midnight).

The context for each trigger event is specified in use cases, and storyboards, which form part of the requirements specification.

Trigger events may be classified as being interaction based, state-transition based or user request based.

Interaction Based Trigger Events can be based on another interaction. For example, the response to a query (which is an interaction) is an interaction based trigger event.

State-Transition Based Trigger Events result from a state transition as depicted in the state transition model for a particular message interaction. For example, the change in status of an order from 'request' to 'fulfillment' is a common state transition. In practice, state-transition based trigger events are frequently encountered.

User Request Based Trigger Events may be based on a user request, such as clicking a mouse. The term 'environmental' is also used. For example, the trigger event that prompts a system to send all accumulated data to a tracking system every 12 h is considered environmental as no human user is involved.

Application Role

An Application role is a collection of communication responsibilities intended to be implemented as a group. Communication responsibilities are identified as the interactions that the system is able to send or receive. Application roles may be specializations of other existing application roles, inheriting the responsibilities of its parent, with additional or more specialized responsibilities added, or they may be the merging together of other application roles acting as components.

From the application role definitions, the reader can identify the purpose for information flow between two healthcare applications and the roles that those healthcare applications play in that exchange.

The application role description sets out what one application does, with respect to information exchange. It lists all of the interactions, sent or received, consequent to one particular trigger event. It is silent about the application functionality behind it – and how this is achieved.

Application roles have responsibilities, which are restricted to sending messages (interactions). Any other responsibilities and actions are outside the HL7 model. The sender role has the responsibility to send a message in response to a trigger event, and the receiver role may have responsibilities to initiate further transactions such as an acknowledgement, error report, response to query etc. These are referred to as **receiver responsibilities**.

The application role is a key element in specifying conformance and for contractual arrangements between users and service providers. It is the intent of HL7 that healthcare systems be able to declare conformance to the HL7 specification by creating an implementation profile that identifies the application roles supported by that implementation. Conformance to an application role means supporting each of the interactions specified.

Typically, one application role supports several interactions. For instance, a query is meaningless unless it includes a response, so the application role for the query questioner requires at least two interactions (query and response) to be supported, and similarly for the query answerer.

The names given to application roles provide one of best ways of finding the transaction sets, already defined, which meet a particular requirement. The naming convention is to state the subject of the interaction (e.g. Residential Address) followed by the application role category.

HL7 uses the following generic terms for application roles:

- Placer: An application that is capable of notifying another application about a significant event, and expects the receiver to take action. For example, a clinical system places and order for a laboratory test.
- Fulfiller: An application that is capable of receiving a request from a Placer application. For example, a laboratory system is to fulfill a an order
- Confirmer: An application that is capable of accepting a request from a Fulfiller application. For example, the laboratory system confirms receipt of order.
- Confirmation receiver: A role implemented by a placer indicating what types of confirmations it accepts. For example the clinical system receives confirmation from laboratory.
- Informer: An application that is capable of notifying another application about a significant event, but does not expect any action on the part of the receiver. Paired with tracker. For example a patient admission system informs laboratory that patient has been admitted.
- Tracker: An application that is capable of receiving information about a significant event, but is not expected by the receiver to perform any action. For example laboratory tracks patients who have been admitted.

In theory, application roles should be helpful to the reader in understanding the business roles and functionality provided by a set of interactions. However, the use of abstract terms, such as manager, tracker, placer and filler, makes this less useful than it might be.

Message Type

A message type is the most precise specification of a message, with explicit constraints about what data elements are sent and what values each data element may have. These constraints should be as tight as possible to minimize any chance of ambiguity.

Message types are derived by the intersection of specific interactions, application roles, and trigger events. The same message type may be associated with any number of application roles and be used in response to many different trigger events. However, an interaction can only ever have one trigger event and one message type.

Interaction Sequence

The precise flow of messages may be represented using a UML sequence diagram, which shows the application roles and the flow of message types between them in sequential order.

Message Wrapper

Whenever domain content (as a payload) is transmitted in the form of messages they use message wrappers, analogous to a letter's envelope. HL7 defines two types of wrapper: a transmission wrapper and a trigger event control act wrapper. Each HL7 Version 3 message typically consists of a transmission wrapper, a trigger event control act wrapper and the domain content.

The **transmission wrapper** includes a unique reference ID for each message instance sent, the precise date and time the message was created and the identity of the sending and receiving systems.

The **trigger event control act wrapper** sits inside the outer transmission wrapper and may include details of a previous interaction, which has triggered this interaction. Different variants of the Trigger Event Control Act wrapper are used for asynchronous messaging and for queries, where the response needs to be coupled with the query (Fig. 16.2).

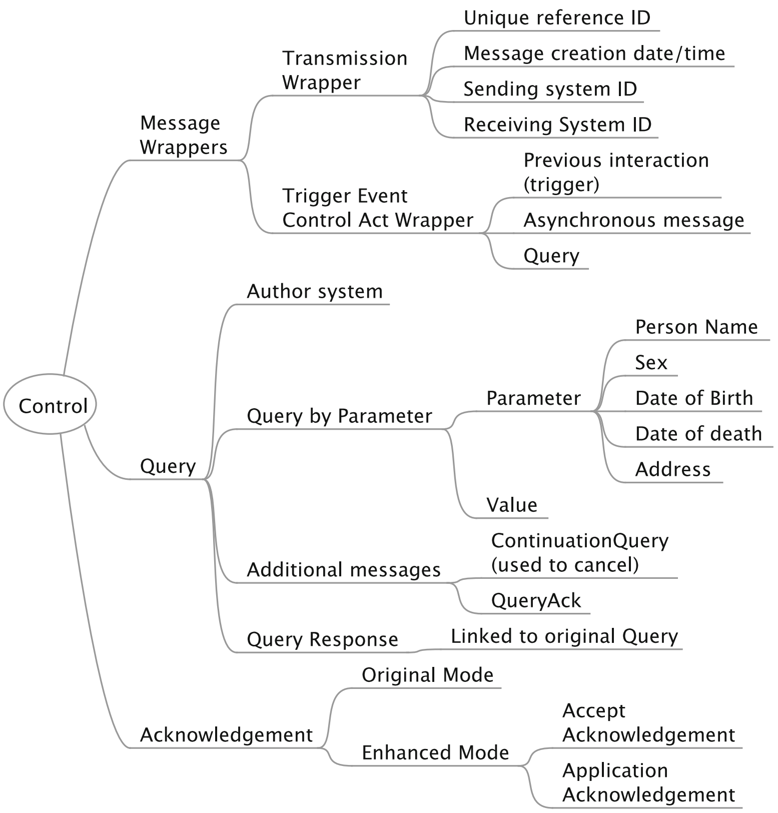

Fig. 16.2 Message communications control

Query

Queries are used to interrogate databases, such as to retrieve patient identification details from a patient master index.

The standard query message is an extension of the control act, using query by parameter. For a simple patient demographics query, the parameters could be patient name, sex, date of birth (and/or death) and address.

The query response is linked to the original query message.

Acknowledgement

Most HL7 transactions involve two or more messages: an originating message and an acknowledgement, in one of two modes – original mode or enhanced mode.

In **original mode acknowledgement** there are just two messages, the first, originating message comes from the sending system and the second, an acknowledgement is sent by the receiver saying whether it was able to process the originating message. Original mode acknowledgement is more straightforward to implement, especially for simple point-to-point interfaces.

Enhanced mode acknowledgement is more complex, but is suited to a multi-hop environment that uses an intermediary such as an interface engine between the sender and the final recipient. In enhanced mode acknowledgement, two separate acknowledgements are sent.

The first, the **accept acknowledgement** is a message indicating whether the receiving system, which could be an interface engine, was able to take custody of the sender's message, but does not indicate whether it was able to process the information contained within it

The second, the **application acknowledgement** indicates whether the final receiving application was able to process the sender's message successfully, indicating end-to-end completion of the transaction.

Safety

Safety is paramount in healthcare. Examples of safety procedures include:

- Acknowledgements sent at both transport level (message received) and application level (message processed)
- Explicit validation by both sender and receiver systems
- Use of automatic patient matching, with fallback of manual matching if not entirely unambiguous

- Routing messages to alternative recipient if not actioned within a specified time (for example if a named recipient is on leave)
- Messages are not removed from a task list until all actions specified have been performed
- If any user edits a message the original is kept unchanged (deletionless messages)
- A full audit trail is maintained.