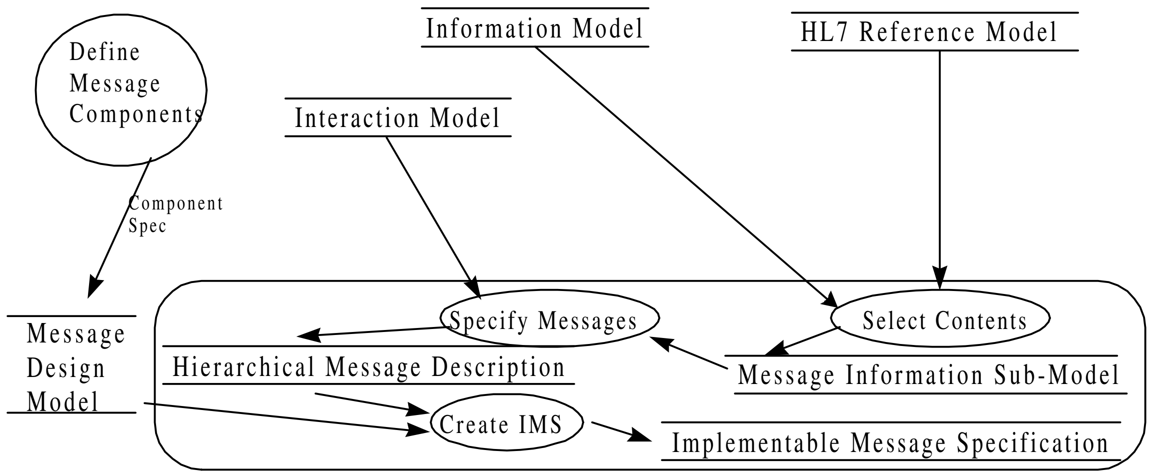
6. Stage IV: Specification (Implementation) - Creating Hierarchical and Implementable Message Specifications

6.1 Overview

In this stage the technical committee actually specifies the format of HL7 messages and relates them to interactions. To do so it goes through the steps illustrated below.



1. It selects a subset of the Reference Model to serve as the information model of the specific domain of interest. This Message Information Submodel (MISM) is directly related to the Information Model produced in Stage II of the process.

2. It builds one or more Hierarchical Message Descriptions (HMD) to describe messages. Each HMD describes one or more messages. Each message may be used to describe the message format that is used for one or more interactions.

3. Specialists work with the Technical Committee to create sections in the Implementable Message Specifications (IMS) that correspond to the message formats described in the HMDs. There will be two Implementable Message Specifications, one for Printable Character Stream messages and one for messages transmitted using OLE and/or CORBA.

6.1.1 The Message Information Sub-Model

The Message Information Sub-Model (MISM) must be a subset of the Reference Model. Its purpose is to create a simplified representation of the contents of the Reference Model by selecting entities that are relevant to a logical group of messages. The MISM will be published with the messages to help users understand their structure and the definition of their data elements.

6.1.2 The Hierarchical Message Description

The Hierarchical Message Description is the primary definition of the contents of a message. In preparing it the technical committee

a) moves from the generality of class to the specificity of objects. If the same class appears several times in a message to represent different objects it is repeated in the description. For example, if a message describes a person who is a patient, another person who is next of kin and another person who is the referring physician, then the Person class may be repeated in the HMD once for each of those objects

b) describes for each object its optionality, its conditionality (conditions under which it may or may not be included in the message), and its cardinality (how many times it may recur in the message)

c) enumerates each attribute of the class that will appear as a data element in each object representing the class (we may send fewer and different attributes of Person when it is describing the referring physician than we do when it is describing the patient)

d) describes for each data element in the object its optionality, its conditionality (conditions under which it may or may not be included in the object within the message), and its cardinality (how many times it may recur in the object)

e) establishes an order in which the object views can be sequenced.

Because several objects in the message may represent substantially different sets of attributes of the same class, we call the representation of an object in the HMD an **object view**.

The HMD is represented as a table whose rows represent object views and data elements. A row that identifies an object view is labeled with the name of the class that it is drawn from A row representing a data element is labeled with the data element name and type.

Each column represents a message description--the complete listing of all the object views and data elements that may appear in the message. A column heading contains one or more Interaction IDs (from the Interaction Model) that are served by messages that have a common format.

The cell at the intersection of an object view row and a column describes its connection to at most one other object view and its optionality, cardinality, and conditionality. The cell at the intersection of a data element row and a column describes its optionality, cardinality, conditionality and the requirement for specific values.

6.1.3 Implementable Message Specification

The final step in message development is creation of the implementable message specification (IMS). This defines the rules for creating the actual message that will pass between systems using HL7. In order to create the IMS, you need to draw on the message design model which defines the rules for creating messages in a particular syntax supported by HL7. There may be multiple message design models, one for each messaging syntax that HL7 will support, e.g., ER7, OLE, CORBA. For each message design model, a different IMS can be created. This final stage of message construction will be highly mechanical. We may be able to automate all or part of it.

6.1.4 Optionality, Cardinality, Conditionality, and Required Value

These terms are used in the following way in Hierarchical Message Definition. This varies slightly from their usage in the information model.

6.1.4.1 Applied to Object Views

The *optionality* of an object view may be:

- **Required**. The object view must appear every time the message description is used for an Interaction (subject to the optionality implied by cardinality and conditionality, below.)

- **Site Specific.** The object view may appear when the message is used. This will be defined on a site-specific bases. See discussion on HL7 Conformance Claims *(needs to be written.)*

- **Not Permitted.** This object view may never occur when the message description is used for an interaction. (This option exists because the object view represented by the row of the HMD may be used in other message descriptions described by other columns of the same HMD.)

The *cardinality* of an object view refers to the number of times it may repeat in the message. It is specified as a pair of numbers specifying the lower bound and the upper bound, as in (1,m). The left number is the lower bound. It may be zero, one, or any other positive integer. When it is zero, the object view may be optional. The right number is the upper bound. It may be one, any other positive integer, or lower case "m." Lower case "m" indicates an unstated arbitrary number of occurrences.

The *Conditionality* of an object view may be

- **Unstated**. The Object View may or may not appear according to the optionality and cardinality rules.

- **A stated predicate**. The object view may not appear in the message unless the predicate is true. If the predicate is true it must, may, or may not appear in the message according to the Optionality and Cardinality stated above. The predicate is of the form "Object View-Data Element" = <list of values>, where the specified data element is an enumerated data type and the values listed are valid for the data type. If the data element that is the named in the predicate is not present in the message the predicate is false.

It is not a valid construct to include a stated predicate with an optionality of Not Permitted.

The *required value* of an object view adapts the message format to specific Interactions. It requires that an optional object view be sent and further specifies the value of an enumerated data element in the object view. A required value is always stated in the row corresponding to a data element. Its usage is described below, under "Applied to Data Elements."

6.1.4.2 Applied to Data Elements

The data elements that are specified as part of an object view may only appear in the message if the corresponding object view appears. When a value appears for a data element it may have a value, it may have a special value called Null, or it may be "not present." When the object view does appear, the optionality, cardinality, and conditionality of each attribute may be further specified.

The *optionality* of a data element within an object view may be:

- **Mandatory**. The data element must be present every time the object view appears (subject to the optionality implied by cardinality and conditionality, below.) It must not have a null value and it must be present.

- **Required**. A position for data element must be present every time the object view appears. It must be present (subject to the optionality implied by cardinality and conditionality, below) but it may have a null value.

- **Site Specific**. A position for data element must be present every time the object view appears. It may be present (consistent with the optionality implied by cardinality and conditionality, below.) Whether is present will be defined on a site-specific basis. If present, it may have a null value. See discussion on HL7 Conformance Claims *(needs to be written.)*

- **Not Permitted**. A position for data element must be present every time the object view appears. A value may never be present (This option exists because the object view represented by the row of the HMD may be used in other message descriptions described by other columns of the same HMD.) If the data element that is the named in the predicate is not present in the message the predicate is false.

The *cardinality* of an data element refers to the number of times it may repeat within an individual occurrence of an object view. It is specified as a pair of numbers specifying the lower bound and the upper bound, as in (1,m). The left number is the lower bound. It may be zero, one, or any other positive integer. When it is zero, the data element may be not present in an occurrence of the object view. The right number is the upper bound. It may be one, any other positive integer, or lower case "m." Lower case "m" indicates an unstated arbitrary number of occurrences.

A valid HMD may not have any intersection of data element row and column in which the optionality of the data element is Mandatory and the lower bound of the cardinality is zero.

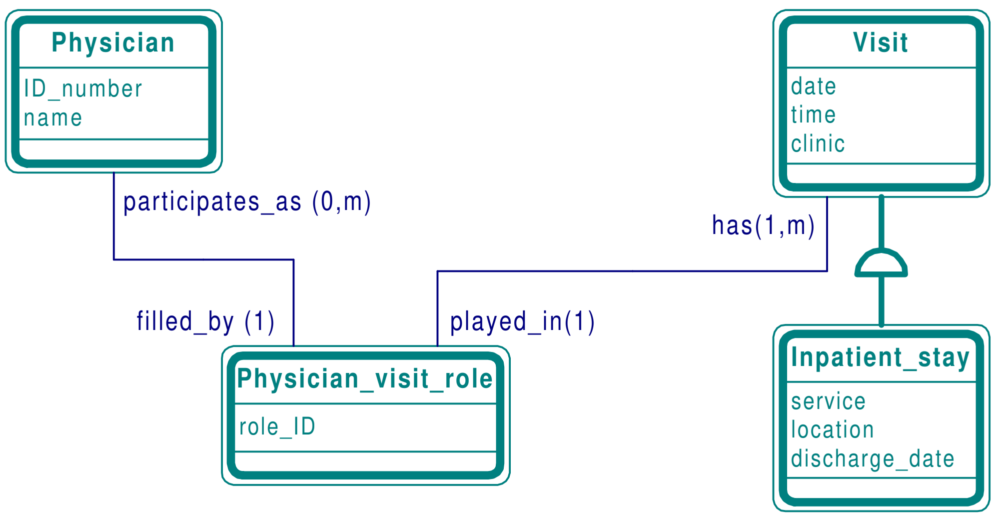
The *Conditionality* of a data element may be

- **Unstated**. The data element may or may not be present in the object view according to the optionality and cardinality rules.

- A **stated predicate.** The data element may not appear in the object view unless the predicate is true. If the predicate is true it must, may, or may not be present in an instance of the object view according to the Optionality and Cardinality stated above. The predicate is of the form "Object View-Data Element" = <list of values>, where the specified data element is an enumerated data type and the values listed are valid for the data type.

It is not a valid construct to include a stated predicate with an optionality of Not Permitted.

The *required value* of an object view adapts the message format to specific Interactions. It requires that an optional object view be sent and further specifies the value of an enumerated data element in the object view.



Consider, for example, the above fragment of an imaginary MISM. Further suppose that role_ID in Physician_visit_role were of an enumerated data type that specifies the role with values for Attending, Admitting, Referring, Senior House Staff, etc. The HMD might include one column for a message that will be associated with the Admit event. That column might include the notation that a value of Attending is required for one occurrence of the Physician_visit_role class. This is equivalent to saying that there must be an attending physician in the Admit message.

Required values for object views are noted at the cell that is the intersection of a column representing a group of Interaction_ids and a row representing a data element within the object view. It contains a value which is one of the enumerated values of the data element.

The inclusion of a required value for an object view implies that the entire object view is required. If the cardinality of the object view connection permits multiple occurrences of the object view, then other occurrences may be in the message, unconstrained as to the value of the specific data element. Multiple required values may be included if multiple occurrences the object view are permitted. Each occurrence of the object view must have one of the stated values.

{It might be better to replace this whole notion with a more general one of business rules--Wes.}

6.2 Creating the Message Information Submodel

6.2.1 Overview

The Message Information Sub-Model (MISM) is created by selecting elements of the Reference Model that are sufficient to describe all the information in a logical group of messages. Using a subset of the entire Reference Model will clarify the information relationships in the messages. Using a single sub-model for the group of messages will decrease the total effort in creating the standard and help its users understand the information relationships among the messages.

Every class, attribute, and connection that appears in the MISM must also appear in the Reference Model. However, the MISM will exclude some classes from the Reference Model. Where it includes a class, it may leave out some of its attributes. Where it includes two classes that have a connection, it may exclude the connection.

Where the MISM includes a connection from the Reference Model it may adjust the cardinality in such a way to reduce the optionality. Specifically

(1,m) can be reduced to (1,1)
(0,m) can be reduced to (0,1)
(0,m) can be reduced to (1,m)
(0,1) can be reduced to (1,1).

However, what was required in the Reference Model cannot be made optional in the MISM.

(1,m) can *not* be reduced to (0,m)
(1,1) can *not* be reduced to (0,1).

Where the MISM includes an attribute from the Reference Model it may be required in the MISM even though it is optional in the Reference Model. However, if it is required in the Reference Model it must be required in the MISM. When an attribute has a cardinality greater than one in the Reference Model it may be reduced in the MISM.

6.2.2 Procedures

The domain Information Model that was created in stage II reflects an understanding of the information that must be sent that was derived from prior analysis of use cases and enhanced through contemporaneous creation of the Interaction Model. It is a primary source of candidate Classes, Attributes and Connections to include in the MISM. However, the scope of the domain Information Model and that of the MISM may be different; several MISMs may document specific groups of messages that are related to a single domain Information Model.

6.2.3 Documentation

The MISM is documented according to the same conventions as those used to describe the Reference Model.

6.2.4 Discussion and helpful hints

There are no exact rules for selecting the contents of an MISM. One strategy is to look at the "largest" message in a group. This would be the Interaction that, according to the use cases, seems to have references to a large number of classes. Technical Committee members may then review other, related Interactions, adding classes as dictated by their use cases.

There is little benefit to making a careful analysis in order to weed out a few data elements from a class. Work can progress faster by allowing them to remain in the MISM and selecting the relevant attributes later, when building the HMD. If, on the other hand, a class with many attributes is included for an incidental reason, requiring only a few of attributes, the Technical Committee may choose to eliminate the unused attributes from the MISM.

There are two benefits to excluding classes and connections from the MISM: (a) it will make the MISM easier to understand for readers of the standard, and (b) various busywork will be avoided if the graphical expression of the MISM fits easily on a page. The cost of excluding information is only that there may be a requirement to add it later, if the committee discovers a need while building the HMD. Since the MISM and HMD are developed during the same stage, it is convenient to make such iterative adjustments in their contents.

Likewise, there are no exact rules for deciding when to use a single MISM and when to have more. Splitting them will make smaller, more manageable graphical and literary expressions. Combining them will help the user understand the information relationships among message descriptions more clearly, and will receive some parsimony in the total length of the specification.

6.2.5 Criteria

6.3 Creating the Hierarchical Message Description

6.3.1 Overview

The HMD is represented as a table whose rows represent object views and data elements. A row that identifies an object view is labeled with the name of the class that it is drawn from and a suffix to distinguish it from other object views drawn from the same class. A row representing a data element is labeled with the data element name and type.

Each column represents a message description--the complete listing of all the object views and data elements that may appear. A column heading contains one or more Interaction IDs (from the Interaction Model) that are served by messages that have a common format.

The cell at the intersection of an object view row and a column describes its connection to at most one other object view and its optionality, cardinality, and conditionality. The cell at the intersection of a data element row and a column describes its optionality, cardinality, and conditionality and required use.

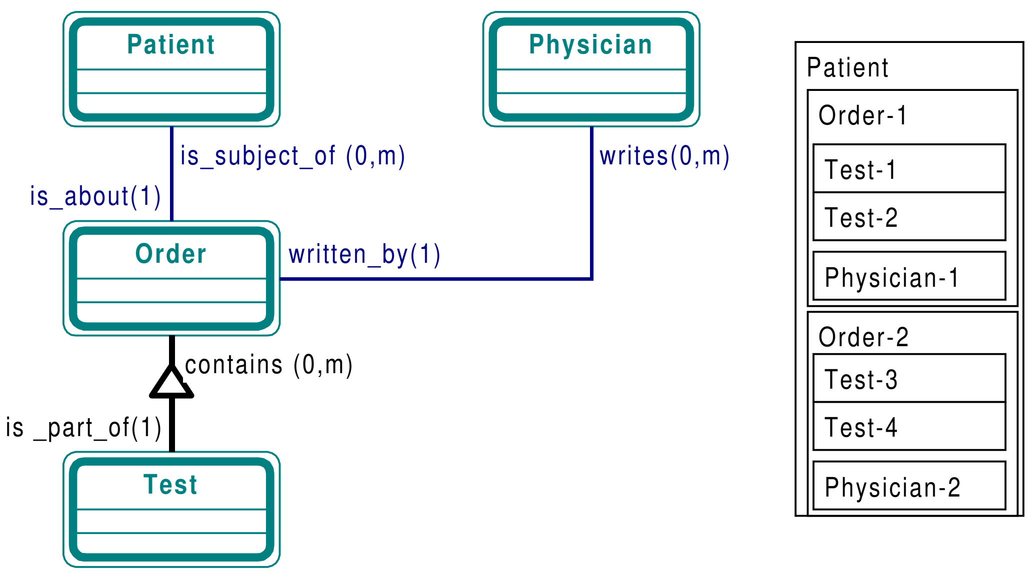
In preparing the Hierarchical Message Description we will answer these important questions

- What object views will be sent in a message?

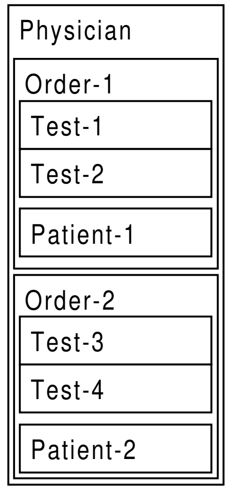
- How are they related to the other object views in the message (by what connection?)
- When will they be sent (always, or are they optional, do they repeat)?
- What attributes of the classes of the MISM from which object views are derived will be sent as data elements for each class?
- In what order will the object views be sent?

The Hierarchical Message Description is a hierarchy in order to portray both sequential relationships among the object views and subordinate relationships. The sequential aspect portrays the order and the subordinate relationship defines the relationship among the object views (as in this is the patient who is the subject of this visit.) In simple messages the relationships may be self-evident. However in most messages the relationships must be clearly shown (as in this is the Person who is the Patient, this is the Person who is the Guarantor, this is the Person who is the Physician who gives primary care to the Patient while that is the Person who is the Physician who will be the Attending physician for this Visit.)

The relationships among object views are most frequently represented in the message by grouping the subordinate objects “within the scope” of the superior object. The physical representation of ‘within the scope’ will vary in the different Implementable Message Specifications, but the logical concept is the same.



The figure above shows an imaginary MISM fragment and one logical interpretation of the subordinate relationships. (Physician-1 is shown “within” Order-1 simply to denote the association between them in a manner consistent with the implementation techniques of the IMSs.) However, the figure below represents another logical interpretation of the subordinate relationships that may be appropriate for a different Interaction.



At the end of the process the HMD, through its subordinate and sequential relationships will be a specific interpretation of the MISM as appropriate for one or more Interactions.

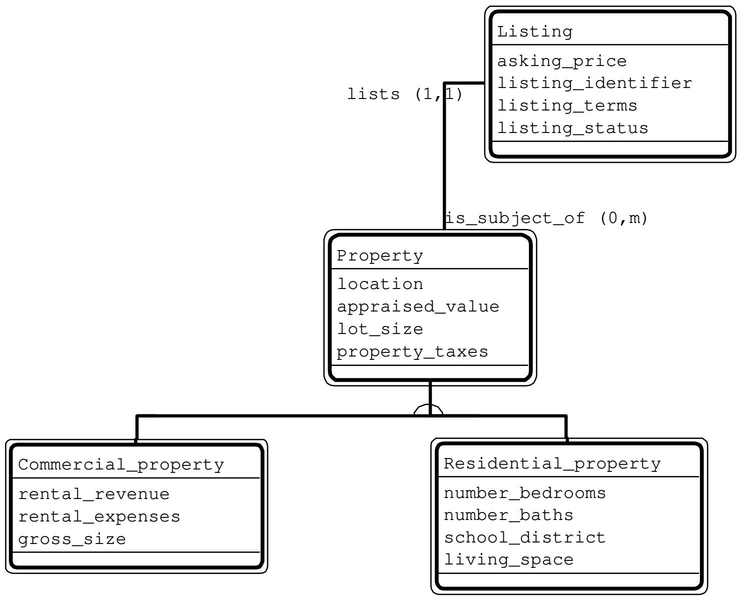
For convenience these "within the scope" relationships will be shown by indentation in Hierarchical Message Description.

So one interpretation of the MISM fragment above might be shown as:

		Patient
is_subject_of	(0,m)	Order
contains	(0,m)	Test
written_by	(1)	Physician

The indentation shows the reader that the written_by relationship is between Physician and Order rather than Test or Patient.

Another issue arises in interpretation of an information model when a connection is shown to a class that is has a gen-spec relationship with another class. Consider the imaginary MISM fragment drawn from the toy model.



Different instance of messages may be describing a Commercial_property, a Residential_property or a Property (not otherwise specified.) These alternatives must be described in the Hierarchical Message Description. This will be done by using the special construct **choice of**.

		Listing
lists	1	**choice of**
		Property
		Commercial_property
		Residential_property

The HMD is produced in three steps:

1. Create a Table of Object Views
2. Create an Adjusted Table of Object Views
3. Create the Hierarchical Message Diagram.

The Table of Object Views shows only the object views (without the data elements.) It is a first draft of the order in which the Object Views will appear in the final message and it has subordinate relationships to describe how the relationships in the MISM should be interpreted for a specific message.

The Table of Object Views can be represented as a table or in a graphical manner.

The Adjusted Table of Object Views is merely a re-ordering of the "first draft" Table of Object Views to optimize certain aspects of the message.

The Hierarchical Message Description is a recapitulation of the Adjusted Table of Object Views with the addition of the data elements.

6.3.2 Procedures

6.3.3 Documentation

6.3.4 Discussion and helpful hints

- Ensure that the attributes from each class in the subset model are represented. Check repetition against the cardinalities in the subset model.

- Pay special attention to situations in which the group of attributes drawn from a class appear multiple times within the table. This can happen when there are multiple instance connections between a class and other classes. Consider whether the multiple connections represent multiple objects (class instances) or repeated uses of the same object. In the second case, you should push these attributes to the top of the table and use a reference number to point to them.

- Data elements need to have the following information recorded for each message profile: xxxx

- You should note that multiple implementable Message Specifications (IMS) can be created for each message. There can be one for each message syntax that HL7 supports. Creation of an IMS is a strictly mechanical process based on the message contents and the message construction rules for the syntax being used.

6.3.5 Criteria

- All attributes from the message sub-model are included as data elements.

- There is a message profile column for each information flow in an Event/Role diagram, unless the project team has clearly labeled two flows as identical.

- Element optionality is recorded for each element and message profile.

- Repeating objects within messages are handled correctly.

6.3.6 Toy Model

Here is the HMD for a new listing.

Condition	Requirement	Connection	Cardinality	Object/Attribute	Data type
				Listing	
				asking_price	NM
				listing_identifier	EI
				listing_status	ID
				listing_terms	FT

Condition	Requirement	Connection	Cardina lity	Object/Attribute	Data type
		lists	1,1	**Property**	
				appraised_value	NM
				location	AD
				lot_size	NM
				property_taxes	NM
				school_district	ST
		choice_of (1of 2)		**Residential_Property**	
				living_space	NM
				number_baths	NM
				number_bedrooms	NM
		choice_of (2 of 2)		**Multi_unit_property**	
				gross_size	NM
				rental_expense	NM
				rental_revenue	NM
				sprinkler_system_ind	ID
		is_made_up_of	1, m	**Rental_unit**	
				number_bedrooms	NM
				unit_designation	ST
				unit_size	NM
		is_mortgaged_to		**Lender**	
		is_role_for		**Stakeholder_L**	
				taxpayer_id	
		choice_of (1 of 2)		**Person_L**	
				person_address	AD
				person_name	PN
		choice_of (2 of 2)		**Organization_L**	
				organization_address	AD
				organization_name	AD
				SIC_number	ST
		created by	1,1	**Real_estate_agent_S**	
				priority_phone	TN
		is_a	1,1	**Person_S**	
				person_name	PN
	role type = “staff”	holds	1,1	**Agency_role_S**	
		staffs	1,1	**Real_estate_agency**	
				agency_listing_identifier	ID
		is_an	1,1	**Organization_A**	
				organization_address	AD
				organization_name	AD

Condition	Requirement	Connection	Cardina lity	Object/Attribute	Data type
				SIC_number	ST
		generalize		**Stakeholder_A**	
				taxpayer_id	ST
	role type = “manager”	staffed_by	1,1	**Agency_role_M**	
		held_by	1,1	**Real_estate_agent_M**	
				priority_phone	TN
		is_a	1,1	**Person_M**	
				person_name	PN

6.4 Creating the Implementable Message Specification for Printable Character Stream Messages

6.4.1 Overview

6.4.2 Procedures

6.4.3 Documentation

6.4.4 Discussion and helpful hints

6.4.5 Criteria

6.4.6 Toy Model

6.5 Creating the Implementable Message Specification for OLE and CORBA Message

6.5.1 Overview

6.5.2 Procedures

6.5.3 Documentation

6.5.4 Discussion and helpful hints

6.5.5 Criteria

6.5.6 Toy Model

7.