

cobas IT 3000

*Host interface manual
version 2.04.00*



©2012 Roche

Roche Diagnostics GmbH
Sandhofer Straße 116
D-68305 Mannheim

Made in Switzerland
www.roche.com

Revision history

Manual version	Software version	Revision date	Main changes
1.0	2.02.01	April 2007	
2.03.04_1	2.03.04	March 2008	ASTM interface
2.03.05_1	2.03.05	July 2008	Enhanced HL7 and ASTM interface
2.03.05_2	2.03.05	October 2008	Added Sample Seen messages to HL7 and ASTM interface
2.03.07 (revision 1)	2.03.07	July 2009	General update, and separate example and system overview sections added.
2.03.08 (revision 1)	2.03.08	February 2010	General update, with corrections.
2.03.09 (revision 1)	2.03.09	July 2010	General update, including instrument time stamp, with corrections.
2.03.10 (revision 1)	2.03.10	September 2010	General update, including confidentiality flags, with corrections.
2.03.11 (revision 1)	2.03.11	April 2011	General update, with corrections. Automatic creation of orderer. Additional sections on message transformations and configuration.
2.03.12 (revision 1)	2.03.12	November 2011	LISDB support for order deletion.
2.04.00 (revision 1)	2.04.0	July 2012	<ul style="list-style-type: none"> Integration of Communication Server 2010. New GUI interface for LISA tasks Updated XSLT stylesheets and configuration descriptions Extended configuration troubleshooting sections, and corrections Added details on automatic order numbers Grouping results options Added warnings on data consistency, patient ID ASCII formats, and single quotes for string literals in XSLT. Updated information on new orderer.

New in this version

Type of change	Summary of changes
Integration of Communication Server 2010	<p>This includes a GUI interface for creating and editing LISA tasks.</p> <p>👁 For details, see <i>Creating the windows service (LISA task)</i> on page A-38, and <i>Creating a LISA task</i> on page A-60.</p>
Grouping results options	<p>👁 For details, see <i>Grouping results</i> on page A-67.</p>
Added warnings on data consistency, and patient ID ASCII formats.	<p>👁 For details, see:</p> <p><i>Danger of patient data inconsistency received from host.</i> on page B-14, <i>Patient ID must be ASCII 7 string</i> on page B-14, <i>Danger of patient data inconsistency received from host.</i> on page C-20 <i>Patient ID must be ASCII 7 string</i> on page C-20</p>
Added warnings single quotes for string literals in XSLT.	

Editor's note

Every effort has been made to ensure that the information contained in this manual is accurate at the time of printing.

Roche Diagnostics Ltd. reserves the right to make any further required changes to software without prior notice. Such changes may not immediately be reflected in this document.

Intended use

This document is intended for the users of **cobas IT 3000**, Version 2.04.00.

Copyright © 2005-2012 Roche Diagnostics International Ltd. All rights reserved.

Trademarks Recognized trademarks

- Roche and the Roche symbol are registered trademarks of the Roche group.
- All other trademarks are trademarks of their owners.
- In this manual as well as in the application, the label cobas IT 3000 is used as an identifier for the Roche Work Area Manager software.

Feedback Every effort has been made to ensure that this guide fulfils its intended purpose as mentioned above. All feedback on any aspect of this guide is welcome and will be considered during updates. Please contact your Roche representative, should you have any such feedback.

Contact addresses

Manufacturer



Roche Diagnostics GmbH
Sandhofer Straße 116
D-68305 Mannheim
Made in Switzerland

Table of contents

Revision history	3
New in this version	3
Editor's note	4
Contact addresses	4
Table of contents	5
Using this manual	7
Conventions in this manual	7
Safety classifications	9
Safety information	9

Introduction **Part A**

1 Introduction to host connections

Typical system architecture	A-4
Starting or stopping a host connection	A-5
How messages are processed	A-6
How a connection starts	A-7
Reading the ASTM or HL7 messages	A-10
Configuring ASTM or HL7 conversion	A-14

2 Setting up the host interface

Overview	A-23
Setting up external order input	A-24
Sending order messages to cobas IT 3000	A-41
Setting the sample ID and order number	A-45
Configuring order result output	A-47
Sending example order results to the host	A-63
Send results on technical validation	A-66
Grouping results	A-67
Using a TCP/IP connection	A-68

HL7 interface **Part B**

4 HL7 protocol

HL7 protocol lower level	B-5
Physical communication	B-5
Minimal Layer Protocol	B-6

5 Description of the HL7 Interface (LISDB)

HL7 communications	B-9
Sending data to cobas IT 3000	B-10
Automatic creation of orderer	B-22
Receiving data from cobas IT 3000	B-25

ASTM interface **Part C**

7 ASTM protocol (LIS2 - A2)

Background to the ASTM protocol	C-5
---------------------------------	-----

Communication processing layers	C-6
ASTM lower layer	C-7
ASTM syntax	C-8
Checksum Calculation / Message Frame	C-11

8 Description of the ASTM Interface (LISDB)

ASTM communications	C-15
ASTM message structure	C-17
Query message	C-18
Order requests (and patient data)	C-20
Automatic creation of orderer	C-29
Test results	C-33
Quality control results	C-39
Sample Event message	C-43

Appendix **Part D**

10 Example XSLT transformations

Transformation of an ASTM message	D-5
Transformation of an HL7 message	D-9

11 The default XSLT transformations

The default XSLT templates	D-17
The HL7 templates	D-18
The default ASTM templates	D-89

12 XML schema definitions

XML schema definitions	D-119
XML schema definition of HL7 and ASTM message structure	D-120
XML schema definition of messages used by cobas IT 3000	D-122

Using this manual

This document describes the online connection of the Roche Diagnostics **cobas IT 3000** to a Laboratory Information System (LIS). It contains a description of the supported message types (e.g. Observation Requests and Observation Results) and their possible fields.

The host interface is typically set up and configured during installation. Therefore, this guide is mainly aimed at system administrators rather than at typical program users.



Incorrect or corrupt data as a result of unauthorized access

Access to your data and the configuration should only be granted to authorized experts. All user actions are logged by the system.

Before starting the configuration of the host interface, ensure that the **cobas IT 3000** was installed successfully. Ensure that all required modules are fully operable and that patient, order and result handling are working correctly.

Every effort has been made to ensure that the information contained in this manual is correct at the time of printing. However, Roche Diagnostics reserves the right to make any required changes to this manual without prior notice within the framework of ongoing product development.





-
- Keep this manual in a safe place where it will not be damaged and will be available at all times.
 - Ensure that this host interface guide can be accessed at any time
-

Conventions in this manual

Symbols and abbreviations are used in this manual to make it easier for you to find and understand the information. The formatting conventions used in this manual are described below.

Symbols The following symbols are used:

Symbol	Meaning
	Cross reference
	Note

Abbreviations The following abbreviations are used:

Abbreviation	Definition
ANSI	American National Standards Institute
GRIPS	Global Repository of Information about Products and Services
GUI	Graphical User Interface. The user screens and dialogs of the computer software.
HIS	Hospital Information System

Abbreviation	Definition
i.e.	that is
LIS	Laboratory Information System
QC	Quality control
SD	Standard deviation

Safety classifications

Safety messages are classified according to ANSI Z535.6. The following classifications are used, according to the level of seriousness of the hazard:

The safety alert symbol by itself (without a signal word) is used to promote awareness to hazards which are generic or to direct the reader to safety information provided elsewhere in the document.

The following symbols and signal words are used for specific hazards:



Warning

Indicates a potentially dangerous situation which, if ignored, may lead to fatal or severe injuries.



Caution

Indicates a potentially dangerous situation which, if ignored, may lead to injuries and/or damage to property.

NOTICE

Notice

Indicates a message not related to personal injury.

According to ANSI Z536.6 there is an additional hazard level: DANGER. Danger indicates a hazardous situation which, if not avoided, will result in death or serious injury. This level is not used in Roche Diagnostics Operator's Manuals. In line with the Roche Diagnostics Product Risk Management Policy, a risk of this degree or level of hazard seriousness is not accepted.

Safety information

Laboratory workflows



Incorrect results due to lack of calibration and quality control

Interrupt the analysis of patient samples if you change the reagent until the instrument has been recalibrated and quality-controlled.



Incorrect results due to expired calibration and quality controls

Perform regular quality controls and calibrations.



Incorrect results due to incorrect entry

Ensure that manually entered data is correct.



Danger of patient data inconsistency received from host.

Do not use this application as a main repository for patient data.

Make sure that the host (LIS/HIS) vendor always sends a unique patient ID for each patient to **cobas IT 3000**.

Any patient demographic changes coming from the host will be updated, no matter if there are results already validated and sent to the host for that specific patient. This may lead to potential wrong validation flags for the affected patient results.

Avoid this kind of change in patient demographic data.



Danger of samples being mixed up due to use of tubes not labeled with barcodes

If possible, always use primary and secondary tubes labeled with barcodes in connection with OMRs and instruct the laboratory staff as to correct handling.



Danger of samples being mixed up due to incorrect assignment of the barcode to the tube

Ensure that assignment of the barcode to the tube is correct during aliquoting. Use the SOP function to alert the laboratory staff to this obligation.



Incorrect results due to lack of knowledge of the Standard Operating Procedures

Use the SOP function of the system to give laboratory staff access to written instructions (SOP) while they work.



Incorrect results due to incomplete patient data

Patient results can only be correctly validated if all of the patient data is stored in the system. In case of incomplete transmission of patient data through the HIS Hospital Information System, tests may have to be repeated.



Misinterpretation of results in case of repetitions

In the case of the transmission of results without timestamp, the designation "Repetition" may be misinterpreted.

In the case of repeated transmission of results, the result gets the status "Repetition". All transmitted results can be displayed with the "Show details" item on the shortcut menu.



Unreliable validation due to unauthorized changes to the validation and calculation rules.

The manufacturer shall not be liable for any consequences whatsoever resulting from subsequent, unauthorized modifications to accepted medical validation and calculation rules included in the system upon customer request.

System Safety

Failure to observe the following safety information may result in incorrect results, data corruption, and data losses.



Incorrect or corrupt data resulting from incorrect operation or the use of wrong components

- Use only computers, monitors, printers, and accessories recommended by the manufacturer.
 - Service your computer regularly (defragment the hard disk; install, run and update antivirus software; and check for system error entries in the event display).
-



Corrupt data due to viruses

Install a firewall, maintain up to date antivirus software and keep your operating system up to date.



Incorrect or corrupt data due to unauthorized access

Access to your data and the configuration should only be granted to authorized experts. All user interventions are logged in the system.

Data security



Data loss

Back up your data at regular intervals (ideally every day).

Maintenance



Data loss or damage to the system due to power failure.

Ensure regular maintenance of the uninterruptible power supply.

Third-party software



Malfunctions and incorrect results due to third-party software

The installation of third-party software that has not been approved by Roche Diagnostics may lead to malfunctions. Do not install any unapproved software.

Introduction

A

- 1 *Introduction to host connections* A-3
- 2 *Setting up the host interface* A-21

Introduction to host connections

Starting and configuring host connections

This section shows how to start and stop a host connections, and explains the basics of host connection configuration. This includes how to use parameters to configure how **cobas IT 3000** reads ASTM and HL7 messages.

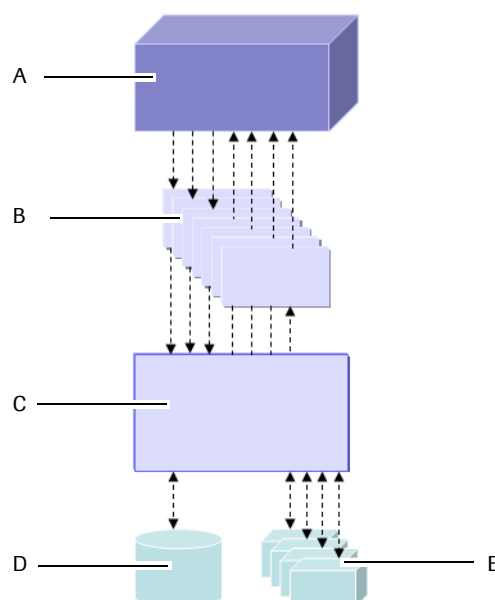
In this chapter

Chapter **1**

Typical system architecture	A-4
Starting or stopping a host connection	A-5
How messages are processed	A-6
How a connection starts	A-7
LISA configuration	A-7
Multiple channels or host connections	A-9
Reading the ASTM or HL7 messages	A-10
Using this manual	A-10
An example conversion (ASTM)	A-11
The XSLT definitions	A-13
Configuring ASTM or HL7 conversion	A-14
Resetting common setting with parameters	A-14
Making extensive reconfigurations	A-15
Finding and editing XSLT stylesheets	A-15
Creating a parameter	A-17
Examples of editing the XSLT	A-18
Adding an alphanumeric order ID to the XSLT for HL7	A-18
Receiving the result report with an alphanumeric order ID	A-19

Typical system architecture

This section gives a brief overview of the system architecture of the communications used between the host and **cobas IT 3000**



- A** The host system. Normally a Hospital Information System / Laboratory Information System / Work Area Manager (HIS / LIS / WAM)
- B** Host connections in LISA tasks. A separate LISA task is started for each host connection. Typically each connection task is implemented as a separate instance of the Communication Server (Host.exe).
- C** **cobas IT 3000**. This implements the business logic that processes data from the host or the instruments, and passes the data on in the required formats to the required destinations.
- D** Database used by **cobas IT 3000** to store patient and order data.
- E** Instruments / analyzers running tests.

Figure A-1 Brief schematic overview of communications

Starting or stopping a host connection

In **cobas IT 3000**, communications with the host are controlled in the **Administration** workplace. A host connection is registered in **LISA Administration** as a task. Check with your Roche Diagnostics Field Service Engineer, or the person who configured your host communication, to be sure which tasks control which host connections. In LISA Administration tasks, you can start or stop a host connection.

► To start a network configuration

- 1 Log in as a user with Configuration privileges.
- 2 Navigate to the screen **Administration > LISA Administration > Tasks**.
- 3 From the list of tasks in the upper part of the screen, select the host communication you wish to start. With the right mouse button, select **Tasks... > Start**.

To stop a task, follow the same process, but select **Tasks... > Stop**.

When you select a command in this screen, **cobas IT 3000** implements it immediately, but the table takes a few seconds to refresh.

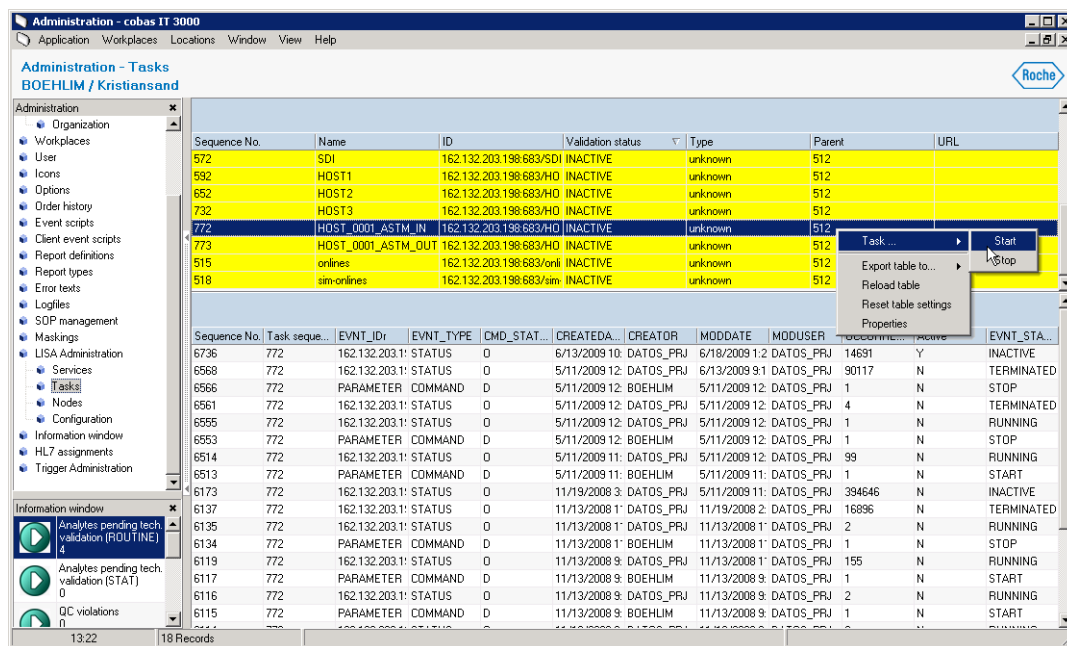


Figure A-2 Administration > LISA Administration > Tasks

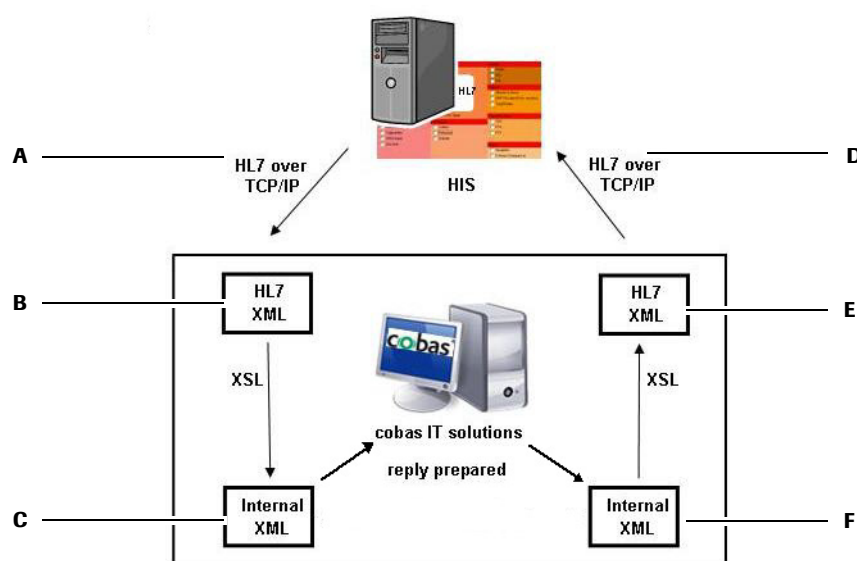
The status of each task is shown by the background color of the line:

Green	Running or active tasks
Red	Task successfully stopped or terminated.
Grey	Task starting or stopping
Yellow	Inactive but available tasks
Blue	Task, selected with mouse-button

How messages are processed

The **cobas IT 3000** can take order data for tests from HL7 and ASTM messages. It can also send the results of the test in HL7 or ASTM format. This manual describes which parts of the HL7 and ASTM protocols **cobas IT 3000** supports, when sending or receiving particular messages.

- 👁 For an example, showing message flow for an order and result messages using HL7 over TCP/IP, see Figure A-3 on page A-6. The same principle applies for other message types (sample event, quality results), other connection types (serial, file transfer) and for other protocols (ASTM).



- | | |
|--|--|
| <p>A The HIS sends HL7 message containing details of requested analytes</p> <p>B On arrival, the message is converted into a generic XML format that describes the structure of the incoming HL7 message.</p> <p>C XSLT conversion of the message into an XML structure that defines the logical structure of the data.</p> | <p>D The XML data is converted to an HL7 message, and then sent to the host.</p> <p>E XSLT conversion of the message into a generic XML format that describes the structure of the required HL7 message.</p> <p>F According to test results or other application processing, the reply data is prepared in an XML structure that defines the logical structure of the data.</p> |
|--|--|

Figure A-3 Example message flow: HL7 interface over TCP/IP

This manual describes the data which can be passed from steps B to C, and from steps F to E in the above diagram: in other words, which HL7 and ASTM fields are supported. The XSL templates are visible in **cobas IT 3000** and can be edited by users and developers: therefore the fields supported may vary slightly from one installation of **cobas IT 3000** to another, and is extensible. This manual describes the data supported in the XSLT files supplied by default in the current version of **cobas IT 3000**.

The stylesheets can be viewed and edited in **Administration > Client > [the name of the current Client] > Configuration** tab.

How a connection starts

This section briefly describes what **cobas IT 3000** does when a LISA task starts a network connection. This aims to help a host developer understand how a connection can be reconfigured, or what is happening at a lower level when a connection starts.

LISA configuration

Each LISA task has a name, given in the *Name* column of the grid in Administration > LISA Administration > Tasks.

👁 For an example, see Figure A-2 on page A-5.

When the user instructs **cobas IT 3000** to start a task, **cobas IT 3000** looks up the task in its LISA manager configuration. This is visible in Administration > LISA Administration > Nodes.

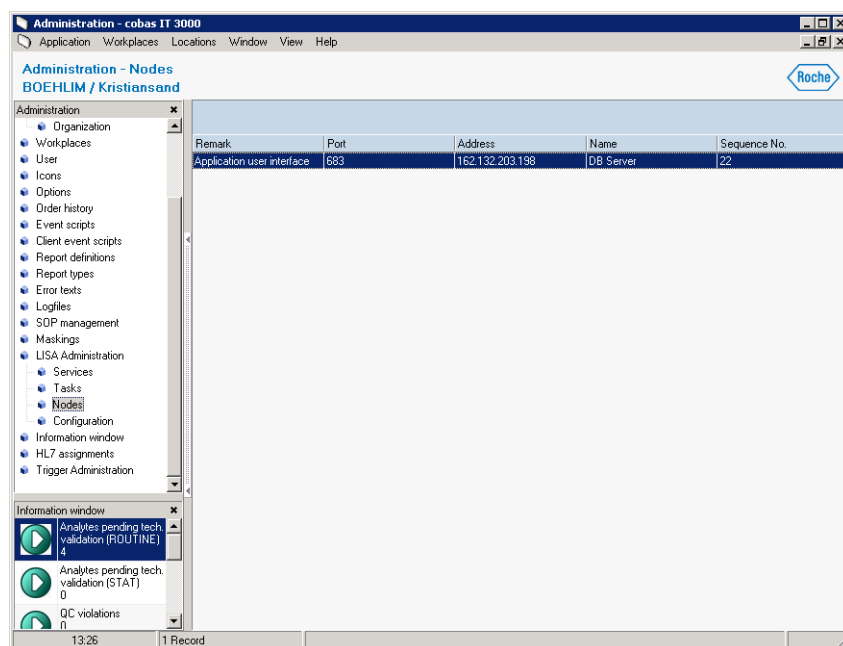


Figure A-4 Administration > LISA Administration > Nodes

Double-click on a node to view its LISA manager configuration.

How a connection starts

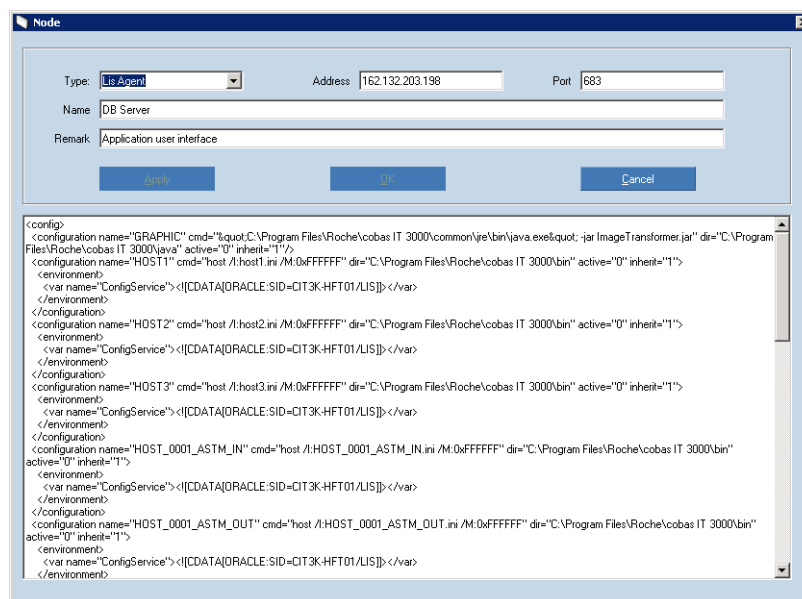


Figure A-5 An example LISA configuration of a node

The LISA configuration tells **cobas IT 3000** how to start the LISA task. For example, a request to start a task that reads in ASTM messages might look as follows.

```
<configuration name="HOST_0001_ASTM_IN" cmd="host /I:HOST_0001_ASTM_IN.ini /M:0xFFFFFFFF"
dir="C:\Program Files\Roche\cobas IT 3000\bin" active="0" inherit="1">
  <environment>
    <var name="ConfigService"><![CDATA[ORACLE:SID=CIT3K-HFT01/LIS]]></var>
  </environment>
</configuration>
```

Figure A-6 An example definition of a LISA task that reads in ASTM messages

When a user starts the task "HOST_0001_ASTM_IN", **cobas IT 3000** looks for the configuration with the name attribute `HOST_0001_ASTM_IN`, and then runs the command specified in the `cmd` attribute. In this case it is

"host /I:HOST_0001_ASTM_IN.ini /M:0xFFFFFFFF"

From this command, **cobas IT 3000** starts the application `host.exe` (the communication server), using the settings in the configuration file `HOST_0001_ASTM_IN.ini`. The other parameters give further settings.

👁 For details of the communication server and the syntax used here, consult your Roche Diagnostics Field Service Representative or the **cobas IT 3000 Service Manual**.

To view or edit the configuration file, see Administration > LISA Administration > Configuration. Protocols, host IP addresses, ports and other set-up information are defined here.

👁 For details of the configuration file settings, consult your Roche Diagnostics Field Service Representative or the **cobas IT 3000 Service Manual**.

The communication server program opens the ports and sets up the connection to the host. In this case, it permits the ASTM messages to be downloaded.

Multiple channels or host connections

A single communication channel (or host connection) can handle all communications between a host and **cobas IT 3000**. However, it is recommended to define a separate communication channel for each type of message, in each direction, each with its own LISA task. This way, any problems or maintenance work only affects part of the system, and debugging and maintenance is easier.

HL7 messages The message types sent over an HL7 connection with a host system normally include some or all of the following. It is best practice to define a separate LISA task for each of these message types. There are standard example configuration (.ini) files to assist with this.

- Order entry, importing order requests from a ward, or updating, replacing or deleting orders.
- Order export to host, sending order requests to another system.
- Result export to the host
- Result import from the host from another HIS or laboratory
- Patient export to the host, sending patient details to another HIS or laboratory
- Patient Administration import or export for transferring details of adding or updating patient data.
- Billing export interface, to host

Although standard practice is to define a separate LISA task for each of the above message types with HL7 messages, it is possible to define other LISA tasks to handle HL7 messages in other ways.

ASTM messages The messages sent by an ASTM connection with a host normally include some or all of the following. It is best practice to define a separate LISA task for each of these message types. There are standard example configuration (.ini) files to assist with this.

- Query message to host regarding samples
- Order message from host with details of tests to be done
- Order result message to host, with details of results of tests done
- Quality result message to host, with details of the results of tests done on quality control samples
- Sample event message to host, with details of where the sample is (for example in an archive), or giving other important information about the processing of the sample

Although standard practice is to define a separate LISA task for each of the above message types with ASTM messages, it is possible to define other LISA tasks to handle ASTM messages in other ways.

Reading the ASTM or HL7 messages

An ASTM or HL7 message goes through these steps when it arrives in the **cobas IT 3000**.

- **cobas IT 3000** converts it to an XML file that shows the structure of the ASTM or HL7 message.
- **cobas IT 3000** performs an XSL transformation that converts it into an XML file that shows the logical structure of the data in the message.

Users or Roche Diagnostics Field Service Representatives can edit this XSLT to meet local requirements. The XSLT stylesheets are stored in **Administration > Client > [Name of Client]**.

- **cobas IT 3000** updates its database from the XML file that shows the logical structure of the data.

When **cobas IT 3000** sends a message to a host, it performs the same steps in reverse order.

Using this manual

This manual describes:

- the ASTM and HL7 fields that are supported by the default interface.
- the elements used in the XML files that describe the logical structure of the data.
- the default ASTM and HL7 fields that correspond to the XML elements.
- the most useful recommendations for localization of the default ASTM and HL7 XSLT stylesheets.

Therefore, you can use this manual:

- to configure a host to use the default interface.
- to guide you when you edit the XSLT stylesheets, to localize the **cobas IT 3000** to your specific requirements.

An example conversion (ASTM)

For example, an incoming ASTM message may contain patient data, in a P record, as follows:

P|1|100000104|10774373||TRÖSEL^BRainer||19871122|M

Figure A-7 Extract from an ASTM message, showing patient demographic data

Conversion to an XML representation

When this message arrives, **cobas IT 3000** converts it to a generic XML format that describes the structure of the ASTM message. It uses the communication server module called “LIS” to perform this conversion, which is configured in the communication task’s configuration (.ini) file. If you need to set up or reconfigure this module, consult your Roche Diagnostics Field Service Representative or the *cobas IT 3000 Service Manual*.

```
<REC ID="P">
  <FIELD NO="2" TEXT="1">
    <DATA> <COMP NO="1" TEXT="1"> </COMP> </DATA>
  </FIELD>
  <FIELD NO="3" TEXT="100000104">
    <DATA> <COMP NO="1" TEXT="100000104"> </COMP> </DATA>
  </FIELD>
  <FIELD NO="4" TEXT="10774373">
    <DATA> <COMP NO="1" TEXT="10774373"> </COMP> </DATA>
  </FIELD>
  <FIELD NO="6" TEXT="TRÖSEL^BRainer">
    <DATA>
      <COMP NO="1" TEXT="TRÖSEL"> </COMP>
      <COMP NO="2" TEXT="BRainer"> </COMP>
    </DATA>
  </FIELD>
  <FIELD NO="8" TEXT="19871122">
    <DATA> <COMP NO="1" TEXT="19871122"> </COMP> </DATA>
  </FIELD>
  <FIELD NO="9" TEXT="M">
    <DATA> <COMP NO="1" TEXT="M"> </COMP> </DATA>
  </FIELD>
```

Figure A-8 Extract from a “raw” XML file, presenting data structured according to the ASTM message

This XML file is then processed by a (user-configurable) XSLT stylesheet. To decide which XSLT file to use, **cobas IT 3000** looks at the “Root name”, i.e. the highest level XML element, of the incoming XML file.

Identifying which XSLT to use

The **cobas IT 3000** system decides which XSLT file to use by looking at the “Root name”, i.e. the highest level XML element, of the file to be transformed. You can match the XSLT file to the XML it used on by looking at the “Root name” column in **Administration > Clients > [ClientName] > Configuration**.

Root name	Used for
ASTM	Used for an incoming ASTM message
HL7	Used for an incoming HL7 message
OrderResponse	Used to create an HL7 acknowledgement message
OrderResults	Used to create a message sending order results, in either HL7 or ASTM.
SampleEvent	Used to create a message about a sample event, such as sample seen or sample archived, in either HL7 or ASTM.
QualityResult	Used to create a message sending the results of a quality control test.

The XSL transformation

For an incoming ASTM order message, the Root name is “ASTM”. The default XSLT file that applies to this Root name contains the following XSL (in part):

```
<xsl:template match="REC[@ID='P']" mode="body">
  <xsl:element name="PID">
    <xsl:attribute name="PATID">
      <xsl:value-of select="FIELD[@NO=$patid]/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="LASTNAME">
      <xsl:value-of select="FIELD[@NO='6']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="FIRSTNAME">
      <xsl:value-of select="FIELD[@NO='6']/DATA/COMP[@NO='2']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="DOB">
      <xsl:value-of select="FIELD[@NO='8']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="SEX">
      <xsl:variable name="sex" select="FIELD[@NO='9']/DATA/COMP[@NO='1']/@TEXT"/>
      <xsl:choose>
        <xsl:when test="$sex='F'">W</xsl:when>
        <xsl:when test="$sex='W'">W</xsl:when>
        <xsl:when test="$sex='M'">M</xsl:when>
        <xsl:otherwise>
          <xsl:value-of select="U"/>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:attribute>
    <xsl:apply-templates select="*" mode="orc"/>
  </xsl:element>
</xsl:template>
```

Figure A-9 Extract of an XSLT file, for transforming the patient data portion of a message

The XML representation of the data’s logical structure

This XSLT file transforms the data from the message into an XML format that shows the logical structure of the data. In this case, as follows:

<PID PATID="10774373" LASTNAME="TRÖSEL" FIRSTNAME="BRainer" DOB="19871122" SEX="M">

Figure A-10 Extract of an XML file showing patient demographic data

From this XML file, **cobas IT 3000** updates its internal data and user display, and updates its database. For this, it uses the communication server module “LISDB”, which is configured in the communication task’s configuration (.ini) file. If you need to set up or reconfigure this module, consult your Roche Diagnostics Field Service Representative or the **cobas IT 3000 Service Manual**.

Other transformations

When **cobas IT 3000** sends messages to the host, there are separate stylesheets for making the necessary transformations in the opposite direction.

There are other stylesheets for the conversion of HL7 messages, and for the conversion of other types of message.

The XSLT definitions

It is possible to reconfigure the XSLT stylesheets that read the ASTM or HL7 messages, so that they parse the messages in the way you wish them to. Any change you make to the stylesheets will over-ride the settings defined in this manual.

This manual provides a description of the default stylesheets, and describes which ASTM and HL7 fields are supported by **cobas IT 3000**. This aims to give a satisfactory result for most installations of **cobas IT 3000**, but each installation may have been reconfigured by service engineers or previous host developers, so you should be prepared for occasional differences.

Configuring ASTM or HL7 conversion

The **cobas IT 3000** reads the data passed in an ASTM or HL7 message into its internal structure, saving the information in the database, and displaying it in the appropriate fields in the user interface.

👁 For full details of how to set up and configure a host interface connection, see Chapter 2 *Setting up the host interface*.

Resetting common setting with parameters

You can use parameters in **Administration > Clients > [Name of Client] > Configuration** which configure certain values in the interface. These parameters tell **cobas IT 3000** to read certain data from specific ASTM or HL7 fields.

For example, it is usually possible to specify which ASTM field **cobas IT 3000** should read the sample ID from: by default this is field 3 of the O record, but it is possible to tell **cobas IT 3000** to read it from field 4 of the O record.

These parameters enable you to set common variations in the ASTM or HL7 messages used in individual installations.

Depending on the precise type of message and your local configurations, the parameters give users the option to define, these options include:

- which field in the patient record holds the patient ID
- which fields in the order record hold the order ID and sample ID
- the names of the sending and receiving applications
- how ACK messages are handled
- other settings that often need to be reset to comply with particular systems

Depending on the precise installation on your site, only some of these may be available, or there may be more parameters offered.

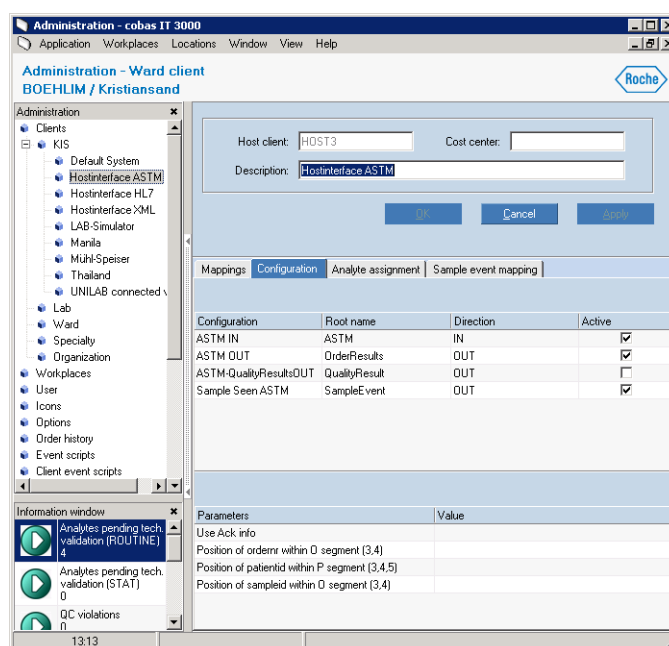


Figure A-11 Configuration of the host interface

► To set interface parameters

- 1 To set interface parameters to define aspects of the interface, navigate to the workplace Administration and select the option **Clients** (or **Location**). Select the host client / location with which you are working.
- 2 Select the **Configuration** tab. The upper grid displays a list of the type of messages that interface can handle.
- 3 Select the type of message you wish to set parameters for. The lower grid displays the configurable parameters.
- 4 In the **Value** box of a parameter, enter the value you wish to set the parameter to. Any values you set here will over-ride the defaults described elsewhere in this manual.
 - 👁 The parameters that each message supports, when using the default templates is described along with the message field definitions in Chapter 5 *Description of the HL7 Interface (LISDB)*, and Chapter 8 *Description of the ASTM Interface (LISDB)*.

If you are comfortable editing XSLT, you can edit the XSLT stylesheets to create new parameters.

👁 For details, see *Creating a parameter* on page A-17.

Making extensive reconfigurations

It is also possible to reconfigure **cobas IT 3000** by editing the XSLT stylesheet that reads the message. This functionality is far more powerful than merely setting a parameter.



How to edit XSLT

To edit the XSLT stylesheets, you should learn at least some basic XSLT and XPath. You can find tutorials and reference information on XSLT and XPath on the internet, for example at <http://www.w3schools.com>. Please note that Roche Diagnostics is not responsible for the content of this or any other external website.

Finding and editing XSLT stylesheets

This section explains how to find and edit the XSLT stylesheets.

► To edit the XSLT stylesheets

- 1 Navigate to the workplace **Administration** and select **Client** (before version 2.03.09, **Location**). Then from the tree select the location or client you are using and the connection.
- 2 Select the tab **Configuration**. In the upper grid, a list of supported messages appears.
 - 👁 For a picture, see Figure A-11 on page A-14.
- 3 Double click the message whose stylesheet you wish to edit. The stylesheet displays in a the **Maintain Configuration** dialog.

Maintain configuration

Host system: **HOST3 Hostinterface ASTM**

Configuration: **ASTM IN** Active: ☒ Queue: **IN_QUEUE** Direction: **IN**

Root name: **ASTM**

Target settings:

Application: **Hospital Information System** Location: **HOST3**

Queue: **IN_QUEUE** Direction: **IN**

Send after (sec): **0** Number of attempts: **5**

Retry after (sec): **10** Wait for Ack. (sec): **0** Priority: **medium**

XSL Code:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!--
=====
* DOCUMENT ELEMENT: xsl:stylesheet
=====
-->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" indent="yes" version="1.0" encoding="UTF-8"/>

  <xsl:param name="ackn">
    <!-- <data name="ackn" type="boolean" comment="Use Ack info" /> -->
    <xsl:value-of select="false()" />
  </xsl:param>

  <xsl:param name="patid">
    <!-- <data name="patid" type="integer" comment="Position of patientid within P segment (3,4) -->
    <xsl:value-of select="4"/>
  </xsl:param>

  <xsl:param name="samplid">
    <!-- <data name="samplid" type="integer" comment="Position of samplid within O segment (3) -->
    <xsl:value-of select="3"/>
  </xsl:param>

  <xsl:param name="ordernr">
    <!-- <data name="ordernr" type="integer" comment="Position of ordernr within O segment (3,4) -->
    <xsl:value-of select="3"/>
  </xsl:param>

  <xsl:template match="/">
    <xsl:element name="OrderRequest">
      <xsl:apply-templates select="*" mode="header"/>
      <xsl:apply-templates select="*" mode="body"/>
    </xsl:element>
  </xsl:template>
</xsl:stylesheet>
```

Figure A-12 The Maintain configuration dialog

4 Confirm that the settings in the upper part of the dialog are correct.

- **Configuration** holds the name for the configuration in the **Configuration** tab.

Example: HL7 Output University Hospital

- Select the **Active** checkbox to show that the configuration is in use.
- In **Root name**, enter the root name of the XML created by the HL7 or ASTM interface.
 - 👁 For details of the root name, see *Identifying which XSLT to use* on page A-12
- From the **Direction** list, select **IN** or **OUT** (direction of data transfer from the point of view of the **cobas IT 3000**).
- In **Location**, enter the LOCATION value in the configuration (.ini) file.
- In **Queue** enter the value assigned the QUEUE parameter in the configuration (.ini) file. This is the Queue for Oracle Alert messages from the database.
 - 👁 If two host communication channels use the same host **Location** or **Client**, they must have different **Queue** values. See *Location and Queue* on page A-17.

Recommended target settings:

- From the **Application** list, select **HIS** or **Hospital Information System**.
- In **Send after (sec.)**, enter "0".
- In **Retry after (sec.)**, enter "10".
- In **Number of attempts**, enter "5".
- From the **Priority** list, select **medium**.

5 The lower part of the dialog holds the XSL used to transform the message. To edit this conveniently, copy and paste all of the XSL into a text editor or XML editor. The main chapters of this manual describe the meanings of the XML elements, and how they relate to HL7 and ASTM fields, and the supported values. Use this information for guidance.

- 6 When you are satisfied with your edits, copy and paste it back into this dialog box, replacing all the previous text. Select **Apply** or **OK**. Test it with some sample messages.



Editing the XSLT overrides the default information in this manual

This manual describes the default templates. The default templates assign the values in the HL7 and ASTM fields to specific elements. However, after a developer has edited a stylesheet, the HL7 and ASTM fields may not be assigned to the default elements. Be careful, and check for local variations.

This manual contains some recommendations for altering or extending the stylesheets in certain circumstances. If you edit the stylesheets, make a written record of your changes for future developers.

- 👁 For a specific example of how to edit the XSLT, see *Examples of editing the XSLT* on page A-18.

Location and Queue

The **Location** value is taken from the LOCATION parameter in the .ini file.

If several host communication tasks use the same **Location**, they must have different **Queue** values. The Queue value is taken from the QUEUE parameter in the .ini file. Two tasks must have a different value for either their the **Location** and **Queue** values.

For example consider a configuration that has the following host communication channels defined.

LOCATION: HOST1	QUEUE: ORDER_IN
LOCATION: HOST1	QUEUE: RESULT_OUT
LOCATION: HOST2	QUEUE: ORDER_IN
LOCATION: HOST3	QUEUE:

You can create another host communication task with the Location `HOST2` and the Queue `RESULT_OUT`, as that would still be unique.

You cannot create another host communication task with the Location `HOST1` and the Queue `RESULT_OUT`, as this is identical to one already used. Behavior in this situation could be unpredictable.

You cannot create another host communication task for the Location `HOST3`, because this has no Queue parameter set. If you want to add another host communication task to `HOST3`, you must give every host communication task in `HOST3` a Queue value.

Creating a parameter

At the beginning of each stylesheet, a number of parameters are initialized. These parameters are presented to users in **cobas IT 3000** in **Administration > Client > [Name of Location] > Configuration**. Users can set the initial values of these parameters without having to edit the XSLT.

- 👁 For details of how to use the parameters, see *Resetting common setting with parameters* on page A-14.

You can edit these parameters, and add new parameters, by editing the XSLT.

The stylesheets describe the parameters in a simple proprietary syntax, which is an extension to XSLT. The **cobas IT 3000** reads this description, and presents parameters to the users according to the definitions. Before the XSL transformation is done on the incoming message, **cobas IT 3000** initializes the parameters to the values the users have set.

The **cobas IT 3000** reads the definition of the parameter from a commented out `<data>` tag, for example:

```
<xsl:param name="patid">␣
  <!-- <data name="patid" type="integer" comment="Position of patientid within PID segment {3
    [external],4 [internal],5 [alternate]}" /> -->␣
  <xsl:value-of select="3"/>␣
</xsl:param>␣
```

Figure A-13 XML definition of patid parameter

The data element has the following attributes.

- **name.** This must match the name attribute of the `xsl:param` element which the definition applies to. The **cobas IT 3000** reads this attribute, and uses it to decide which `xsl:param` element to assign values to.
- **type.** The data type of the parameter. This can be integer, string, boolean, or in principle any type supported by `xsl:param` in XSLT, so long as it can be entered into the field in the **Configuration** tab. This information is for reference only, and no check is made on the data.
- **comment.** This sets the text displayed in the Configuration tab to explain the parameter to the user.

The `<xsl:value-of select="...">` element sets the default value of the parameter. This is overridden by any value the user sets.

Examples of editing the XSLT

This section provides examples of how you might implement some functionality using the XSLT, following the procedure described above.

👁 For a general description of how to edit the XSLT, see *Making extensive reconfigurations* on page A-15.

Adding an alphanumeric order ID to the XSLT for HL7

If you wish to use an alphanumeric order ID, as well as or instead of the default numeric order ID (or “order number”), this is possible in **cobas IT 3000**. However, the default templates do not support this. Neither do the default parameters give you an option to implement this. However, there is an XML element and attribute, `<ORC HOST_ORDER_ID>`, that takes an alphanumeric order ID.

First, decide in which field to send the alphanumeric order ID. In this example, we will pass it in an HL7 message, in ORC-3, the third field of the ORC segment, which is defined in the HL7 standard as “Filler Order Number”. For example, 98765MM4FM, in ORC below:

```
MSH|^~\&|HIS|FAC1|CIT3K|FAC2|20100624124858||ORM^O01|20100624124858093|P|2.3|||ER|ER||8859/1
PID|1||2010994||Testus^SUN2^J||196303241225|F
ORC|NW|2010994|98765MM4FM|||^^^^^R
OBR|1|2010994|^^^^K-S||199808101444|||A|||SER
OBR|2|2010994|^^^^NA-S||199808101444|||A|||SER
OBR|3|2010994|^^^^LH||199808101444|||A|||SER
```

Figure A-14 Order sent with alphanumeric order ID

► **To add support for sending an alphanumeric order ID to cobas IT 3000**

- 1 Navigate to Administration > Client > Configuration, and select HL7_IN.xsl. The XSLT used to parse the incoming message will appear in a dialog box.
- 2 Copy and paste the XSLT content to an XML editor or text editor, such as Notepad. Save a backup copy.
- 3 Search the XSLT and find the start of the element ORC, as follows:


```
<xsl:element name="ORC">
```
- 4 Inside the ORC element, insert the following attribute:


```
<xsl:attribute name="HOST_ORDER_ID">
  <xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='1']/@TEXT" />
</xsl:attribute>
```
- 5 Save the edited file, then copy and paste the XSLT back into cobas IT 3000.
- 6 Test your host interface thoroughly before implementing it live.



Receiving the result report with an alphanumeric order ID

To receive the alphanumeric order ID from cobas IT 3000, edit HL7_OUT.xsl in the same way.

► **To add support for receiving an alphanumeric order ID from cobas IT 3000**

- 1 Navigate to Administration > Client > Configuration, and select HL7_OUT.xsl. The XSLT used to parse the outgoing message will appear in a dialog box.
- 2 Copy and paste the XSLT content to an XML editor or text editor, such as Notepad. Save a backup copy.
- 3 Search the XSLT and find the start of the element ORC, as follows:

```
<xsl:template match="ORC" mode="orc">
  <xsl:element name="REC">
    <xsl:attribute name="ID">ORC</xsl:attribute>
```

Note that if you are not using the default template, the exact location might be different. If so, find the equivalent point in your XSLT.

- 4 Inside the ORC element, insert the following attribute:

```
<xsl:element name="FIELD">
  <xsl:attribute name="NO">3</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@HOST_ORDER_ID"/>
  </xsl:attribute>
</xsl:element>
```

- 5 Save the edited file, then copy and paste the XSLT back into the cobas IT 3000.
- 6 Test your host interface thoroughly before implementing it live.



This example produces a result output like the following:

```
MSH|^~\|CIT3K|FAC1|HIS|FAC2|20100624125246||ORU^R01|36937|P|2.4|||ER|NE||8859/1|
PID|1||2010994||Testus^Sun2|||W|
PV1|1|||||||||||||||||||||||||||||||||
ORC|1|2010994|98765MM4FM|||||20100624124858|
OBR|1|2010994||K-S^101|||20100624125243|
OBX|1||K-S^101^|2^No·normal·range·predefined!||·-·^|||F|||20100624125243|^|BOEHLIM^24.06.10^^|
```

Figure A-15 Result message with alphanumeric order ID

Setting up the host interface

How to set up and configure a host interface connection

This chapter tells you how to set up the host interface over HL7 or ASTM using the default templates and settings.

In this chapter

Chapter **2**

Overview	A-23
Setting up external order input	A-24
Before you start	A-24
Configuring cobas IT 3000 parameters	A-25
Creating the host client	A-25
Configuring general data parameters	A-26
Defining the analytes	A-28
Configuring cobas IT 3000 administration options	A-29
Defining the Host client	A-31
Check your host client	A-31
Creating the XSLT configuration	A-31
Mapping analyte codes	A-33
Adding INI file to database	A-34
Creating the windows service (LISA task)	A-38
Adding the order task to the LISA tasks	A-38
Starting the new LISA task	A-39
Setting up host interface logging (optional)	A-39
Sending order messages to cobas IT 3000	A-41
Sending a new HL7 order	A-41
Sending a new ASTM order	A-42
Troubleshooting	A-43
Setting the sample ID and order number	A-45
Configuring order result output	A-47
Before you start	A-47
Configuring cobas IT 3000 parameters	A-48
Creating the host client	A-48
Configuring cobas IT 3000 parameters	A-49
Defining the analytes	A-50

Defining the result reports	A-52
Configuring cobas IT 3000 administration options	A-54
Defining the Host client	A-55
Check your host client	A-55
Creating the XSLT configuration	A-55
Mapping analyte codes	A-57
Adding INI file to database	A-57
Creating a LISA task	A-60
Adding a new LISA task	A-60
Starting the LISA order results task	A-61
Setting up host interface logging (optional)	A-62
Sending example order results to the host	A-63
Troubleshooting	A-64
Send results on technical validation	A-66
Grouping results	A-67
Using a TCP/IP connection	A-68
System files	A-68
Setting up the HL7 connection	A-69
Setting up the ASTM connection	A-69
Network port configuration	A-69
TCP/IP server	A-70
TCP/IP client	A-70
Data framing and flow protocols	A-70
HL7 configuration	A-71
HL7 flow control	A-71
ASTM configuration and flow control	A-71
Buffer Size	A-71
Further protocols supported	A-72

Overview

This chapter describes how to set up the **cobas IT 3000** to receive order requests from a host system, and return test findings. The **cobas IT 3000** application can be integrated into almost any host capable of communicating via HL7 or ASTM. You to configure the mapping from the message data to **cobas IT 3000** internal data, and vice versa. In addition, there are parameters to configure multiple connections over a wide variety of communication protocols, for details talk to your Roche Diagnostics service personnel or see the *cobas IT 3000 Service Manual*.

The **cobas IT 3000** uses two modules to control host communication:

- The LIS module, which converts the HL7 or ASTM message into an XML format, that describes the content of the fields and components of the message.
- The LISDB module, which uses a configurable XSLT transformation to convert the XML output of the LIS module into an XML format that describes the logical structure of the data. It then uses this to update the **cobas IT 3000** database.

For incoming messages, the **cobas IT 3000** uses these modules in the order LIS - LISDB. For outgoing messages, it uses them in the opposite order LISDB - LIS. You can configure the mapping of the data using XSL, which provides a very powerful transformation functions.



Other modules and protocols are also available.

Some **cobas IT 3000** installations may alternatively process host messages with other modules or transformation methods, and may support other protocols. Talk to your Roche Diagnostics service personnel or see the Chapter 41 *Configuring connections and protocols* for further details.

-
- 👁 For a fuller description of how the **cobas IT 3000** communications work, see *How messages are processed* on page A-6, and *How a connection starts* on page A-7.
 - 👁 For a fuller description of how **cobas IT 3000** processes messages, see *Reading the ASTM or HL7 messages* on page A-10.

Setting up external order input

This section describes how to configure the **cobas IT 3000** to receive orders from the host by file transfer.

In ASTM, the message consists of H, P, O and L records. In HL7, the message implements the OML^O21 or ORM^O01 event.

HL7 event	Interface event	Comment
OML^O21 (ORM^O01)	Order Request	Send in MSH-9.
Note: "oh-two-one", or "oh-zero-one".		The 5th character of HL7 code is a capital letter O.

Table A-1 HL7 event used for external order input

Use this message for sending new orders, and for updating, replacing, or deleting orders.

To set up external order input from an host:

- Make sure you have the necessary information before starting.
- Check or configure **cobas IT 3000** it can receive orders.
- Configure the **cobas IT 3000** administration options for receiving orders.
- Define the host client that handles communication with the host.
- Set up and start the LISA task (a Windows service) that the host client uses.
- Optionally, set up logging for the communication.

Before you start

Before you start, make sure you know:

- The code the host sends for each:
 - Specialty (default HL7 PV1-4, not sent by default in ASTM)
 - Orderer (default HL7 PV1-3 or ORC-10, or ASTM O-17)
 - Analyte or profile (default HL7 OBR-4, or ASTM O-5.4)
 - Priority of orders (default HL7 OBR-7, or ASTM O-6)
- The internal codes your **cobas IT 3000** site uses for each:
 - Specimen
 - Analyte
- The XSLT stylesheet to use for your host connection for incoming messages. By default this is HL7_IN.xsl or ASTM_IN.xsl, available on GRIPS or from your Roche Diagnostics service personnel.

👁 You can also copy them from Chapter 11 *The default XSLT transformations*.

If you use an edited stylesheet, discuss carefully what changes have been made with previous service personnel. The XSLT files provide very powerful reconfiguration options, so if edited, may require some care. In particular check:

- The sample splitting mode, and sample create mode that are supported on that site.
- Example HL7 and ASTM messages that the site supports.

Follow their advice on these points, and any other points they raise. In other respects, follow the set-up instructions here.

- The configuration file (.ini file) to use on your site for the input LISA task. If you do not have this, you can create one from the instructions given later.

Configuring cobas IT 3000 parameters

This section describes how simply to configure **cobas IT 3000** parameters to process external orders. These settings might already be configured on your system, but check them carefully. To set the parameters:

- Define the Host ID of the host client you will use.
- Make sure the **cobas IT 3000** general data parameters are configured.
- Define the analytes that the host can order.

Creating the host client

You need to define an “host client” inside **cobas IT 3000**. This holds the definition of the connection to the host, and contains an “Host ID” that other parts of the application use to identify that connection to that host. For now, just define the host client’s “Host ID”. Complete the full details of the host client configuration later.

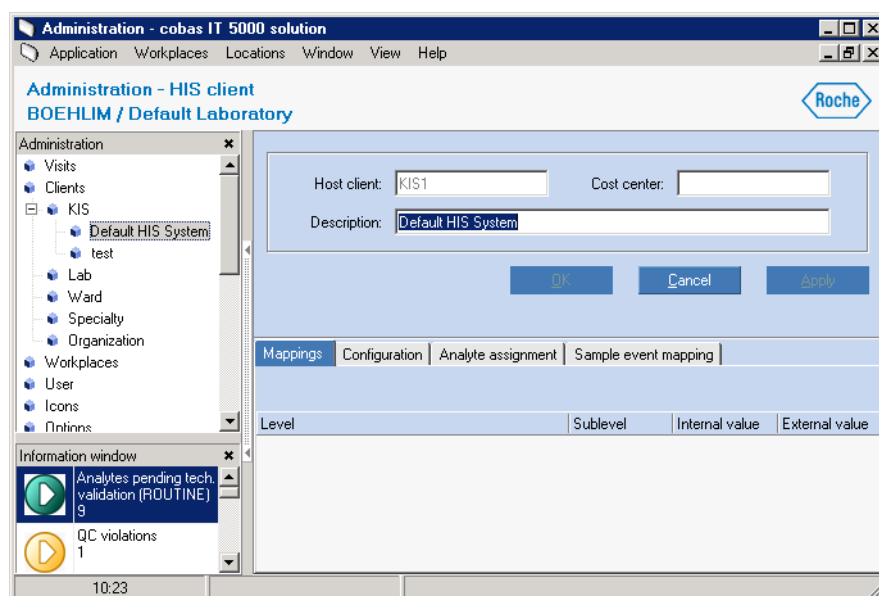


Figure A-16 Creating a host client and Host ID

If no host client is configured on your system, create one as below.

► Create a host client

- 1 Navigate to **Workplaces > Administration > Locations > KIS**.
- 2 In the main window, from the context menu, select **Insert new Host location**. The **Host location** dialog opens.

Field or control	Enter	Comment
Host client	Short code to identify the connection.	Often called the “Host ID”. This is used in: <ul style="list-style-type: none"> the Location parameter in the configuration (.ini) file. the Output type or Host system field of the Report result configuration dialog. And many other places.
Description	The full name or description of the client.	
Cost center	Leave empty	

Table A-2 Entries in the **Host location** dialog

We can define the actual details of the Host client connection later. We need the “Host ID” field for the general configuration of **cobas IT 3000**.

- 3 Navigate to **Workplaces > Parameter > Organization**. In the main window, from the context menu, select **Insert new organization** or **Edit organization**. The **Organization** dialog opens.

Field or control	Enter	Comment
Organization code	Short code to identify the organization.	You cannot edit this later.
Host system	Select the host client that handles communication with this organization.	
(other fields)	Optional, but recommended.	Fill the details of the host or other host organization.

Table A-3 Entries in the **Organization** dialog

This defines the Host ID for the client that handles connections to the organization. Next we need to make sure that the **cobas IT 3000** has sufficient general parameters set to handle incoming data.

Configuring general data parameters

In this section we check the parameters that define who can give orders, what levels of priority can be assigned to orders, as well as specimen types and medical specialties supported.

► Configure cobas IT 3000 parameters

- 1 Navigate to **Parameter > Specialties**. From the context menu, select **Insert new specialty**, or **Edit specialty**.

Field or control	Enter	Comment
Code	Short code	ID shown in the GUI. For example, SURG
Designation	Specialty	Name shown in the GUI. For example Surgery

Table A-4 Entries to the **Specialty** dialog

Field or control	Enter	Comment
HIS code	Code sent by host, defining the specialty.	By default, sent in HL7 PV1-4. Not sent by default in ASTM.
Ward ID		Leave blank

Table A-4 Entries to the **Specialty** dialog

- 2 Navigate to **Parameter > Orderer**. From the context menu, select **Insert new orderer** or **Edit orderer**, to open the **Orderer** dialog. This defines who can create orders.

Field or control	Enter	Comment
Code	Short code ID shown in the GUI.	For example, WA1. This cannot be edited later.
Orderer	Orderer's name shown in GUI.	For example Ward 1
Organization	Select from list.	Taken from Parameter > Organization .
Host code	Code sent by host, defining the orderer.	By default, sent in HL7 PV1-3 or ORC-10, or ASTM O-17.
Specialty list	Select specialties	Open Specialty dialog from the context menu in the lower list.
(other fields)		Are optional (for order input)

Table A-5 Entries to the **Orderer** dialog

- 3 Navigate to **Parameter > Specimen**. From the context menu, select **Insert new specimen**, or **Edit specimen**, to open the **Edit specimen** dialog.

Field or control	Enter	Comment
Code	Short code ID shown in the GUI.	For example, 01. This cannot be edited later.
Name	Name shown in GUI.	For example, Serum
Analytes list		You cannot assign analytes here. Assign them later in Parameter > Analytes / reference ranges . (Step 2)
(other fields)		Are optional (for order input)

Table A-6 Entries to the **Specimen** dialog

- 4 Navigate to **Parameter > Priorities**. From the context menu, select **Insert new priority** or **Edit priority**, to open the **Priority** dialog. It is recommended to have at least an "R" priority for "Routine" orders, and an "S" priority for STAT ("Short Turn-Around Time") orders.

Field or control	Enter	Comment
Code	Short code ID, used internally, and sent by host.	For example, R. This cannot be edited later. By default, sent in HL7 OBR-7, or ASTM O-6.
Name	Priority name shown in GUI.	For example, Routine

Table A-7 Entries to the **Priorities** dialog

Field or control	Enter	Comment
Priority	An integer code for priority. 1 has the highest priority.	
Color	The color the GUI will display for the order.	
Active	Select checkbox	
(other fields)		Are optional (for order input)

Table A-7 Entries to the **Priorities** dialog

This sets up the general parameters needed for handling incoming orders. Next you must set the analytes that **cobas IT 3000** supports.

Defining the analytes

You need to define the analytes that **cobas IT 3000** supports.

► Define the analytes

- 1 Navigate to **Parameter > Analyte groups**. From the context menu, select **Insert new analyte group** or **Edit analyte group**, to open the **Edit analyte group** dialog.

Field or control	Enter	Comment
Code	Short code ID shown in the GUI.	For example, CC. This cannot be edited later.
Name	Name shown in GUI.	For example, Clinical Chemistry.
Analytes list		You cannot assign analytes here. Assign them later in Parameter > Analytes / reference ranges . (Step 2)
(other fields)		Are optional (for order input)

Table A-8 Entries to the **Edit analyte group** dialog

- 2 Navigate to **Parameter > Analytes / reference ranges**. From the context menu, select **Insert new analyte** or **Edit analyte**, to open the **Analytes / reference ranges** dialog.

Field or control	Enter	Comment
Analyte no.	Internal ID number for analyte.	For example, 103. This cannot be edited later.
Analyte	Full name of analyte.	For example: Chloride.
Abbreviation	Internal abbreviation for analyte	For example: CL
Analyte group	Select from list.	Taken from Parameter > Analyte groups .
Specimen	Select from list.	Taken from Parameter > Specimen .
Module	Select LIS.	
Reference ranges list		Optional for order input, but recommended.

Table A-9 Entries to the **Analytes / reference ranges** dialog

Field or control	Enter	Comment
(other fields)		Are optional (for order input)

Table A-9 Entries to the **Analytes / reference ranges** dialog

Repeat this step for each analyte that the host uses.

- 3 If you are using profiles, navigate to **Parameter > Profiles**. From the context menu, select **Insert new Profile** or **Edit profile**, to open the **Profiles** dialog.

Field or control	Enter	Comment
Profile code	Short code ID shown in the GUI.	For example, Electro. This cannot be edited later.
Profile name	Profile name shown in GUI.	For example, Electrolytes
Module	Select LIS .	
Analyte list	Select analytes	Open Assign analytes dialog from the context menu in the lower list.
(other fields)		Are optional (for order input)

Table A-10 Entries to the **Profiles** dialog

Configuring cobas IT 3000 administration options

This section describes how simply to configure **cobas IT 3000** administration options to process external orders. These settings might already be configured on your system, but check them carefully.

This example assumes that your host sends each order with a Sample ID for the sample and a separate Order ID for the order. Some sites use a different configuration. We can edit this configuration later to support different configurations.

► Configure cobas IT 3000 administration options

- 1 Navigate to **Administration > Options > SAMPLES**.

Create an option **SAMPLE_SPLITTING_RULE** if it does not already exist. (Context menu, **Insert new entry. Option group** dialog.)

Field or control	Enter	Comment
Item	SAMPLE_SPLITTING_RULE	
Content	3	This indicates that cobas IT 3000 expects to receive the sample ID from the host.
Group	SAMPLES	
Description	1:(Sample ID = Order ID plus a 2-digit specimen code). 3: (Sample ID sent by host.)	
(other fields)		Are optional (for order input)

Table A-11 Entries to the **Administration > Options > SAMPLES > SAMPLE_SPLITTING_RULE** dialog

Other values for the sample splitting and sample create parameters are possible, and we will consider them later.

👁 For details on sample create rules and sample splitting rules, see page *Setting the sample ID and order number* on page A-45.

- 2 If you have set SAMPLE_SPLITTING_RULE=3, navigate to **Administration > Options > SAMPLES**.

Create an option SAMPLE_CREATE_RULE if it does not already exist. (Context menu, **Insert new entry. Option group** dialog.)

Field or control	Enter	Comment
Item	SAMPLE_CREATE_RULE	
Content	3 This indicates that cobas IT 3000 expects to receive the sample ID from the host.	Integer code for the sample create rule. Default value is 3.
Group	SAMPLES	
Description	Default = 3 if SAMPLE_SPLITTING_RULE is 3. 0: Sample ID = Order ID. 1: Sample ID = Order ID + Specimen. 3: Sample ID must be provided by the host or the sample will be rejected. 4: Sample ID = uses form prefix + Order ID (could be more than one sample).	
(other fields)		Are optional (for order input)

Table A-12 Entries to the **Administration > Options > SAMPLES > SAMPLE_CREATE_RULE** dialog

Other values for the sample splitting and sample create parameters are possible, and we will consider them later.

- 3 Navigate to **Administration > Options > Log**.

Create an option TRACE_LEVEL if it does not already exist. (Context menu, **Insert new entry. Option group** dialog.)

Field or control	Enter	Comment
Item	TRACE_LEVEL	
Content	40	
Group	log	
Description	Trace level for LISDB messages. 0 = OFF, 10 = ERROR, 20 = WARNING, 30 = INFO, 40 = DEBUG.	
(other fields)		Are optional (for order input)

Table A-13 Entries to the **Administration > Options > LOG > TRACE_LEVEL** dialog

Set this option to 0 (= OFF) before going into a production environment.

- 4 Navigate to **Administration > User**. Expand the tree to show [operator's user name] > **Rights > KIS > [host ID]**.

In the main window, select the row marked **Read patient data (SEL_PAT)**, and from the context menu, select **Assign**.

Also assign these rights in the same way:

- **Insert patient (INS_PAT)**
- **Delete patient (DEL_PAT)**
- **Update patient (UPD_PAT)**

Defining the Host client

Earlier we defined the Host ID and the organization it referred to. We left the full definition of the Host client connection to later. Now is the time to fill this in.

Check your host client

Earlier, we defined a host client to handle the connection to the host. This is defined in **Workplaces > Administration > Locations > KIS**, and referenced in **Workplaces > Parameters > Organization**, in the field **Host system**. Check these are set.

👁 For details of we created this, see *Creating the host client* on page A-25.

The **cobas IT 3000** handles messages to or from that host according to the settings in the Host client configuration.

Creating the XSLT configuration

When messages arrive in **cobas IT 3000**, they are automatically transformed into an XML file that describes the content and structure of the original HL7 or ASTM message. The **cobas IT 3000** needs to transform this into another XML format that it needs to update its database. To do this it uses an XSLT stylesheet. You need to define which stylesheet to use for incoming HL7 or ASTM messages.

👁 For an example of a transformation, see Chapter 10 *Example XSLT transformations*. For schema definitions (XSD) of the XML, see Chapter 12 *XML schema definitions*.

Before you start

Make sure you have the default stylesheet for your system. The default stylesheets are available on GRIPS or from your Roche Diagnostics service personnel.

HL7_IN.xsl	For incoming HL7 orders
ASTM_IN.xsl	For incoming ASTM orders

Table A-14 Default XSLT stylesheets

👁 You can also copy them from Chapter 11 *The default XSLT transformations*.

Your site may already use a bespoke stylesheet. If you use an edited stylesheet, discuss carefully what changes have been made with previous service personnel. In particular check:

- The sample splitting mode, and sample create mode that are supported on that site, and that the stylesheet expects.
- Example HL7 and ASTM messages that the site supports.

The method for creating the XSLT configuration is the same for an edited as for a default stylesheet.

► To create the input XSLT configurations:

- 1 Make sure that you have the XSLT file (by default HL7_IN.xml or ASTM_IN.xml) to hand, and it is open and ready for you to cut and paste from.
- 2 Navigate to **Workplaces > Administration > Locations > KIS > [Your host client] > Configuration**.
- 3 From the context menu, select **create new configuration**. To view or edit a pre-existing configuration, double-click it.

The **Maintain configuration** dialog opens.

- 4 Enter/select the following :

Field or control	Content	Comment
Configuration	Enter a name	Example: HL7_IN
Active	Select	
Root name	HL7 or ASTM	The root name of the XML input. 👁 For a full list of root names, see <i>Identifying which XSLT to use</i> on page A-12.
Queue	A unique ID string. This must be different from other configurations in this Host client.	This is the Queue for Oracle Alert messages from the database. This must match the QUEUE parameter in the configuration (.ini) file.
Direction	IN	Direction of data transfer

Table A-15 Values for the upper part of the **Maintain configuration** dialog.

Target settings:

Field or control	Content	Comment
Application	Hospital Information System	Instructs cobas IT 3000 that this configuration is for host communications.
Queue	A unique ID string. This must be different from other configurations in this Host client.	This is the Queue for Oracle Alert messages from the database. This must match the QUEUE parameter in the configuration (.ini) file.
Send after (sec.)	0	
Retry after (sec.)	10	
Wait for Ack. (sec.)	5	
Location	Select your host client	
Direction	IN	Direction of data transfer
Number of attempts	5	
Priority	Medium	Some sites might have a special configuration for urgent orders.

Table A-16 Values for the **Target settings** part of the **Maintain configuration** dialog.

- 5 Copy and paste (Ctrl+C and Ctrl+V) the content of the XSLT file (HL7_IN.xml or ASTM_IN.xml, or whatever) into the text field at the bottom of the dialog. Make sure your cursor is at the very beginning of the text field before you paste the XSL content!
- 6 Choose **OK**.
- 7 If you receive the error message "**Unimplemented or unreasonable conversion requested**", your file contains too many characters. In this case, delete all empty spaces before the tags:
 - Place the cursor in the text field.
 - Press Ctrl+R.
 - In **Text to find**, type two empty spaces.
 - Make sure the field **Replace with** is completely empty.
 - Click **OK**. The XSLT is saved and the dialog closes.

This tells **cobas IT 3000** how to process the message from the client. When a message arrives, **cobas IT 3000** looks at:

- client name or **Location**, in this case the host client we defined earlier.
- The **Application**, which must be **HIS** for host connections.
- The **Direction**, in this case **IN** (from host to **cobas IT 3000**).
- The **Queue**, which is the Oracle queue, and distinguishes this configuration from other configurations with the same details.

From this information, **cobas IT 3000** decides which configuration to use when parsing the message.

Mapping analyte codes

You must map the analyte codes used by the host to the codes used by the **cobas IT 3000**.



Note

Before you can map analyte codes, you must create the analytes in **Parameter > analytes / reference ranges**.

► To map analytes:

- 1 Navigate to **Administration > Locations > KIS > [Your host client] > Analyte assignment**
- 2 From the context menu, select **insert new mapping**.
The **Create new analyte mapping dialog** appears.
- 3 In **Description**, enter the analyte abbreviation.
- 4 From the **Internal Analyte** list, select the relevant analyte.
You can alternatively select a profile from **Internal Profile**.
- 5 In **External analyte (In)**, enter the analyte code received from the host. (Or profile code.)
- 6 In **External analyte (Out)**, enter the analyte code that should be sent to the host (usually this is the same as in **External analyte (in)**). (Or profile code.)
- 7 Click **OK**.

8 Repeat steps 2-7 for all analytes or profiles that can be requested from the host.

This tells **cobas IT 3000** the meaning of the codes sent from the host.



The internal and external mapping must be unique

The mapping from an external code to an internal code is a one-to-one relationship. Each incoming external code must map to a unique internal code. Each internal code must map to a unique outgoing external code.

Adding INI file to database

The **cobas IT 3000** uses configuration (often called .ini or initialization) files to define parameters and DLLs used by the communication server when setting up the host connection.

Make sure you have a configuration (.ini) file prepared. If you do not, create one using this example and edit it match your site, according to the following instructions.

*Example input configuration
(.ini) file*

```

/*****
* OrderHL7.ini
*****/

[GENERAL]
Applications=HOST

[HOST]
Connections=LIS,ORDER

[HOST-LIS]
DllName      = LIS
# read HL7 input from file
# Portconfig = 7,Input,Output,Time
PortConfig   = 7,c:\data\in\*.hl7,,10
BaseProtocol = 204,1000,0,0
#Protocols   =
Parser       = MSXML
Encoding     = ISO-8859-1
Destination  = HOST-ORDER
MessageType  = HL7
TraceFile    = C:\data\log\ordershl7.log

[HOST-ORDER]
DllName      = LISDB
AutoDelete   = 1
Application  = HIS
Location     = KIS1
Driver       = OCIOA10
Database     = COBAS-SERVER01/LIS
NLS_NUMERIC  = .,
Queue        = IN
Encoding     = ISO-8859-1
WriteThreadTimeout = 90000
ReadThreadTimeout = 90000
ReConnectThreadTimeout = 90000

```

What this does The first section of the .ini file

```
[GENERAL]
Applications=HOST
```

says that this file defines an application that it names HOST.

The next section

```
[HOST]
Connections=LIS,ORDER
```

says that the HOST application uses two modules, that it names LIS and ORDER.

The next section:

```
[HOST-LIS]
DllName          = LIS
...
Destination      = HOST-ORDER
```

says that the HOST-LIS module is a LIS.dll module. A LIS.dll module reads an HL7 or ASTM message, and transforms it into an XML file that describes the structure and data of that message. The configuration (.ini) file defines some setup parameters for this LIS module, which we will consider below, and then passes its output to the module HOST-ORDER.

The next section:

```
[HOST-ORDER]
DllName          = LISDB
```

says that the HOST-ORDER module is a LISDB.dll module. A LISDB.dll module reads the output from a LIS.dll (or similar) module, transforms it according to an XSLT file, and then writes the output to a database. The details are given in the other parameters, which we will consider below.



Editing the configuration (.ini) file

Points to bear in mind while editing the configuration (.ini) file.

- The configuration file does not support tabs. You must only use spaces for whitespace.
- You cannot add comments in the middle of a line. You can only comment out a whole line, by starting it with a hash sign (#).
- The last line is not read in. Therefore, make the last line of the file:
end of file

► Editing the .ini file

- 1 Use the example above to start with.
- 2 If necessary, edit the header comment to the name of the file, for example OrderASTM.ini.
- 3 Edit the values for the LIS module.

Parameter	Value	Comment
PortConfig	The full filepath to your input files. If necessary, correct the file type tag to .astm or .dat.	You must create this folder if it does not exist. The *.hl7 ending means it reads any file with the .hl7 tag.

Table A-17 Parameters for the LIS module

Parameter	Value	Comment
MessageType	HL7 or ASTM	Can handle one kind of message or the other, but not both.
TraceFile	Complete filepath.	For debugging, LIS.dll writes all HL7 or ASTM messages to this file.
(Other parameters)	Can be left as the example for now.	

Table A-17 Parameters for the LIS module

If necessary, you can configure a TCP/IP port or other kind of connection, later. To do so, change the `PortConfig` parameter, and check the values of the `BaseProtocol` and `Protocols` parameters.

👁 For more details see *Using a TCP/IP connection* on page A-68.

4 Edit the values for the LISDB module.

Parameter	Value	Comment
Application	HIS	Tells cobas IT 3000 to process this as a host connection.
Location	Host ID of your host client	Workplaces > Administration > Locations > [your Host client] . Take the value in the Host client field.
Queue	Defines the value of the Oracle queue. Distinguishes the configuration from others with the same Application and Location.	This must match the value in Workplaces > Administration > Locations > [your Host client] > Configuration > [select the configuration] > Maintain Configuration dialog > Queue .
Database	The login details of your cobas IT 3000 database.	This is the same as the value for Database in the jboss.ini file in Workplaces > Administration > LISA Administration > Configuration .
(Other parameters)	Can be left as the example for now.	

Table A-18 Parameters for the LISDB module

You can now add the configuration (.ini) file to **cobas IT 3000**.

► Adding the configuration (.ini) file

- 1 Navigate to **Workplaces > Administration > LISA Administration > Configuration**.
- 2 From the context menu, select **Insert new configuration**. The **Create** dialog opens.

- 3 In **ID** and **Name**, enter the name of your file, for example, “OrderHL7.ini” or “OrderASTM.ini”.
- 4 In **Description**, explain that this file is used to receive order messages from the host. (optional)
- 5 Copy and paste the content of the configuration (.ini) into the text field at the bottom of the dialog.
- 6 Click **Save**, and close the dialog. You have set up the configuration file.

Creating the windows service (LISA task)

There must be a Windows service that listens for messages coming in. The **cobas IT 3000** does this by starting a “LIS Administration task” or “LISA task”, defined in **Workplaces > Administration > LISA administration**. Often the “LISA task” does this by listening on a TCP/IP port, but in this example, we set up a service to receive messages by file transfer. The “LISA task” regularly polls a specified directory for ASTM or HL7 files, and if it finds any, passes them into **cobas IT 3000**.

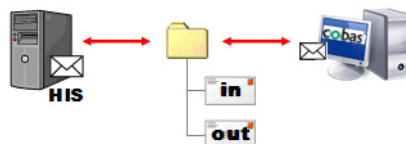


Figure A-17 File-based message transfer

The service is implemented by starting the program `host.exe` with certain parameters, as described below.

LISA tasks launch the parts of the application specified in the INI files. You must create a task which activates the HL7 interface and starts checking for new messages (requests) from the host.

Adding the order task to the LISA tasks

If there is no LISA task that starts your host interface, create one in **Workplaces > Administration > LISA Administration > Tasks**.

Note that in version 2.03.12 and before, LISA tasks were defined with the command-line program `lisamgr`. This may still be needed for legacy systems.

► Creating or editing the LISA task definition

- 1 Go to **Workplaces > Administration > LISA Administration > Tasks**. If there is a definition for your host interface, double click on it. Otherwise to create a new task, from the context menu select **Add new task**.
- 2 In the dialog, edit the following as necessary.

Attribute	Value	Comment
LISA server	The location of the database where the LISA task definitions are stored. Usually localhost.	This cannot be edited later.
External ID	Enter an ID for the task.	This is displayed in the LISAMGR application. This cannot be edited later.
Task name	Enter a name that you can use to identify this task.	This is displayed in LISA tasks.

Table A-19 Creating or editing a LISA task

Attribute	Value	Comment
Command	This is a command passed to the operating system. Enter the filepath of the communication server program, host.exe. Add the parameters In this example: %CIT5K_CS3K5K%\host.exe /I:OrderHL7.ini /M:0FFFFFFF	/I: tells host.exe to load the named .ini file. Enter the name of the .ini file, in Workplaces > LISA Administration > Configuration . /M:0FFFFFFF sets the maximum log level.
Directory	The location of the bin directory of the cobas IT 3000 installation.	
Autostart	Select this to start the task automatically when cobas IT 3000 starts.	
Inherit environment	Select this.	Inherits the environment variables and settings from cobas IT 3000 .

Table A-19 Creating or editing a LISA task

Starting the new LISA task

Before you can start sending orders from the host, you must start the new LISA task.

► To start the HL7 task

- 1 Navigate to **Workplaces > Administration > LISA Administration > Tasks**.
- 2 Right-click your mouse onto the task in the table and select **Task > Start**.

The task is started and the system starts checking for incoming messages (requests) from the host.

When the communication server receives a message, it processes it according to the the configuration (.ini) file, and then writes it to the database.

Setting up host interface logging (optional)

You can set detailed debugging for the communication channel, using the log4j functionality. Make sure that **Administration > Options > Log > TRACE_LEVEL** is set.

► To create a debugging task

- 1 Navigate to **Workplaces > Administration > LISA Administration > Configuration**.
- 2 From the context menu, select **Insert new configuration**.
- 3 The **Create** dialog opens.
- 4 In **ID** and **Name**, enter "OrderHL7.xml", or the name you gave to your order configuration (.ini), but with the tag ".xml".

- 5 In **Description**, explain that this file defines the log4j debugging of “OrderHL7.ini”, or whatever you called the configuration file.
- 6 Copy and paste the contents of the file below into the text field at the bottom of the dialog. Edit the file, so that it shows the log file you wish to write to.

```
<?xml version='1.0' ?>
<log4j:configuration debug='false' xmlns:log4j='http://jakarta.apache.org/log4j/'>
<appender name='DEFAULT' class='org.apache.log4j.RollingFileAppender'>
  <param name='File' value='c:/data/log/OrderHL7.log' />
  <param name='MaxFileSize' value='50MB' />
  <param name='MaxBackupIndex' value='20' />
  <layout class='org.apache.log4j.PatternLayout'>
    <param name='ConversionPattern' value='Trace %d{ISO8601} [%t] %-5p %c %x - %m\n' />
  </layout>
</appender>

<root>
  <priority value='DEBUG' />
  <appender-ref ref='DEFAULT' />
</root>
</log4j:configuration>
```

Figure A-18 The log4j file

- 7 Click **Save**, and close the dialog. Log information will be written to the specified file.

Sending order messages to cobas IT 3000

After you have configured your order input host interface, test it with some sample messages.

Sending a new HL7 order

If you configured an HL7 interface, send a test HL7 order, similar to the following.

```
MSH|^~\&|KIS1|||||ORM^O01|201103071545330001||||ER|ER|
PID|||444444||Coray^Ruth||19301213|W|
PV1|||WA1|SURG|
ORC|NW|8763523||||R||20110307154523|WA1||||01|
OBR||635236||102||20110307154500||||A
OBR||635236||104||20110307154500||||A
```

► To edit the example order

- 1 For convenience, make sure that the datetime (20110307154 in the example) is set to the current date and time, in the format YYYYMMDDHHMMSS[ssss], everywhere it occurs.

- n the PID segment, that either:

- 2 Edit the PID segment, so that either:

- The patient ID, PID-3, 444444 in the example, has not yet been used.

Check the value is unique using the search function in **Workplaces > Routine > Overview samples**.

or:

- All the patient details match a patient already in **cobas IT 3000**.

Search for patients in **Workplaces > Routine > Patients**.

- 3 In PV1-3 enter an orderer, and in PV1-4, a specialty, already defined in **cobas IT 3000**.

Search for these in **Workplaces > Parameter > Orderer**, and **Workplaces > Parameter > Specialties**.

- 4 Enter the same orderer in ORC-10, as entered in PV1-3.

- 5 In ORC-15, enter the specimen ID.

- 6 In ORC-2 enter an order ID that has not yet been used.

Check the value is unique using the search function in **Workplaces > Routine > Overview samples**.

- 7 In OBR-2 enter a sample ID that has not yet been used.

Check the value is unique using the search function in **Workplaces > Routine > Overview samples**.

- 8 In OBR-4, enter an analyte that exists in **cobas IT 3000**.

Search for these in **Workplaces > Parameter > Analytes / reference ranges**.

► **To send a test order**

- 1 Once you have prepared the HL7 order, make sure that your LISA task for HL7 orders is running. See **Workplaces > Administration > LISA administration > Tasks**.
- 2 Put your HL7 file in the input directory defined in your configuration (.ini) file. Wait ten seconds. The **cobas IT 3000** will read it and then delete it from the input directory.
- 3 Navigate to **Workplaces > Routine > Day list**. The order is in the list of orders.

Sending a new ASTM order

If you configured an ASTM interface, send a test order, similar to the following.

```
H|\^&|||ASTM-Host||||PSM|P|20110308125500
P|1||444444||Coray^Ruth||19301213|W|||||||Diabetes||||201103081255
00|||||||Urology
O|1|876400010801|^102^104|R|201103081255000|201103081255000|||A
|||01|HOST|||||||O
L|1|F
```

Make sure you are using the default XSLT templates, called ASTM_IN.xml. If not ask for advice from the person who edited your templates, or from someone who can read them. This example, and the following instructions, may not apply.

► **To edit the example order**

- 1 For convenience, make sure that the datetime 201103071545 is set to the current date and time, in the format YYYYMMDDHHMMSS[ssss], everywhere it occurs.
- 2 In the P segment, that either:
 - The patient ID, P-4, 444444 in the example, has not yet been used.
Check the value is unique using the search function in **Workplaces > Routine > Overview samples**.
 - or:
 - All the patient details match a patient already in **cobas IT 3000**.
Search for patients in **Workplaces > Routine > Patients**.
- 3 In O-5, enter analytes that exists in **cobas IT 3000**.
Search for these in **Workplaces > Parameter > Analytes / reference ranges**.
- 4 In O-3, make sure that the first ten digits are an order number (or “order ID”) that has not yet been used. Note that if using the default ASTM templates, only the first ten digits of this field are the order number.
Make sure also that the whole of O-3 has not been used as a sample ID.
Check the order number and sample ID are unique using the search function in **Workplaces > Routine > Overview samples**.
- 5 In O-16, enter the specimen ID.
- 6 In O-17 enter an orderer already defined in **cobas IT 3000**.
Search for orderers in **Workplaces > Parameter > Orderer**.

► To send a test order

- 1 When you have edited and prepared an order, make sure that your LISA task for ASTM orders is running. See **Workplaces > Administration > LISA administration > Tasks**.
- 2 Put your ASTM file in the input directory defined in your configuration (.ini) file. Wait ten seconds. The **cobas IT 3000** will read it and then delete it from the input directory.
- 3 Navigate to **Workplaces > Routine > Day list**. The order is in the list of orders.

Troubleshooting

For debugging, make sure that **Administration > Options > Log > TRACE_LEVEL** is set to 40. Reset this to 0 after troubleshooting.

You can see log information of the communication in **Workplaces > Administration > Communication messages**. The successful message reads, going across the columns: **Pass - HIS - [Host ID] - [Queue]- Order Request - IN - Order Request OK**. Double click on the message, and the **Information** tab reads **OrderRequest OK**, and the lower panel has tabs showing:

- **Raw Content:** The original ASTM message
- **Old content:** The first XML content displaying the content of the ASTM message.
- **XML content:** The final XML content that **cobas IT 3000** uses to update the database.

This communication log often has useful information in case something has gone wrong. There are further suggestions here.

The order is still in the input directory.	Check: <ul style="list-style-type: none">• The correct LISA task is running.• The input directory matches the value in the .ini file in PortConfig.• The file type of the message in the directory matches the file type specified the .ini file in PortConfig.• Check the permissions on the input directory: can cobas IT 3000 read and delete files there?
The LISA task won't start.	Check: <ul style="list-style-type: none">• The LISA task is correctly named and defined in the list of LISA tasks. Look at the entry in Administration > LISA administration > Nodes.• The database is correctly defined in the LISA task. Look at the entry in Administration > LISA administration > Nodes.
In Administration > Communication messages , message reads "Fail", and the Message name is "ASTM". Double click the message, and the Information tab reads "Not supported root name <ASTM>".	If using HL7: In the configuration (.ini) file make sure MessageType = HL7. If using ASTM: Make sure the Queue parameter in the configuration (.ini) file matches the Queue field in the Host client definition in Administration > Locations > [Host client] Configuration > [config] .

Table A-20 Order input, troubleshooting suggestions.

In **Administration > Communication messages**, the message reads “Fail”, and the **Message name** is “HL7”.

Double click the message, and the **Information** tab reads “Not supported root name <HL7>”.

The order has disappeared from the input directory, but has not appeared in **Administration > Communication messages**.

The order has disappeared from the input directory, but has not appeared in **Administration > Communication messages**.

The log file specified in the log4j config has a message similar to:

```
ERROR BCNormalHostThread - HIS Exception
<HostAppLayerError> on handle of <HL7>
```

pr:

```
ERROR BCNormalHostThread - HIS Exception
<HostAppLayerError> on handle of <ASTM>
```

In **Administration > Communication messages**, the message reads “Pending” or “Fail”, but there is no log information in the tabs **Error logs** and **Trace logs**.

In **Administration > Communication messages**, the message reads “Pending” or “Fail”, and in the tabs **Error logs** and **Trace logs**, there is a message:

LIS-10005: Orderer does not exist

In **Routine > Day list**, or in other views on the order, some or all analytes are missing.

In **Administration > Communication messages**, the message reads “Pending” or “Fail”, and in the tabs **Error logs** and **Trace logs**, there is a message:

LIS-10055: There is no specimen with code: xx.

If using ASTM:

In the configuration (.ini) file make sure `MessageType = ASTM`.

If using HL7:

Make sure the `Queue` parameter in the configuration (.ini) file matches the **Queue** field in the Host client definition in **Administration > Locations > [Host client] Configuration > [config]**.

Check the ASTM or HL7 input file went into the correct input directory.

Check that the `Destination` parameter is correctly defined in the configuration (.ini) file.

Remember that the whole name of the destination must be specified, with a hyphen: APP-MODULE, for example:

```
Destination = HOST-ORDER
```

In **Administration > Options > Log** set `TRACE_LEVEL = 40`.

Check the value of the orderer in (HL7) ORC-10 or (ASTM) O-17 exists in **Parameters > Orderer**.

There could be other similarly useful messages here.

In **Administration > Communication messages**, even if the message reads “Pass”, the tabs **Error logs** and **Trace logs** may have a useful error, such as:

Analyte or profile not configured

In **Administration > Options > SAMPLES** set `SAMPLE_SPLITTING_RULE = 3`.

Or, if you are not using this setting, check your sample splitting rule settings, and the format of your order ID in (HL7) ORC-2, or (ASTM) O-3.

Table A-20 Order input, troubleshooting suggestions.

Setting the sample ID and order number

The example order input request we gave above assumed that the host sent a sample ID and an order number. However, on many **cobas IT 3000** sites, the host sends only one or the other of these. In these cases, **cobas IT 3000** needs to create an order number or sample ID.

Sample splitting rules

Make sure you know the policy your **cobas IT 3000** site has for its order number and sample ID. You can then set the parameter `SAMPLE_SPLITTING_RULE` in **Administration > Options > SAMPLES**.

Sample splitting rule	Functionality
1	The host sends a sample ID (by default in OBR-2 or O-3). The cobas IT 3000 splits this number: <ul style="list-style-type: none"> the last two digits become the specimen ID, the other digits become the order number. This is the default setting, but not often used.
3	The cobas IT 3000 creates the sample ID according to the rule set in the parameter <code>SAMPLE_CREATE_RULE</code> .

Table A-21 `SAMPLE_SPLITTING_RULE`

Sample creation rules

If `SAMPLE_SPLITTING_RULE=3`, then you must also set `SAMPLE_CREATE_RULE` in **Administration > Options > SAMPLES**.

Sample create rule	Functionality
0	The host sends an order number. The cobas IT 3000 uses the order number as the sample ID.
1	The host sends an order number. The cobas IT 3000 creates a sample ID = order number + Specimen ID.
3	The host sends: <ul style="list-style-type: none"> a sample ID, otherwise cobas IT 3000 rejects the sample. an order number, otherwise cobas IT 3000 creates an internal order number (an integer incrementing by 1 for each new order.) This is the default setting.
4	The host sends an order number. The cobas IT 3000 creates a sample ID = form prefix + order number.

Table A-22 `SAMPLE_CREATE_RULE` settings

Note that if the host actually does supply a sample ID, this will override the settings in `SAMPLE_CREATE_RULE`.

To tell **cobas IT 3000** which HL7 or ASTM fields hold the sample ID and order number, use the parameters in the **Administration > Locations > KIS > [Host client] > Configuration tab > [your order request input configuration] > Parameters** window.

Setting the sample ID and order number

(Default values of parameters)	HL7	ASTM
Sample ID	OBR-2	O-3
Order number	ORC-2	O-3 (first 10 digits only)

Table A-23 Default fields from which **cobas IT 3000** reads the sample ID and order number



To tell cobas IT 3000 not to read in a sample ID or order number

To tell **cobas IT 3000** not to read in the sample ID or order number, set the value of the parameter to zero.



Recommended sample rules

It is recommended to use sample-create-rule 1 and sample splitting rule 3 for cobas IT 5000 systems.

It is recommended to use sample-create-rule 3 and sample splitting rule 3 for cobas IT 3000 systems.

Logout and restart LISA task

After setting these options, you must log out of **cobas IT 3000** completely, including closing the login screen, and restart the LISA task for host communication.

If the host does not send an order number

If the host does not send an order ID, **cobas IT 3000** creates an order number. The automatic order number is in the range permitted for the host client, and not more than 100 numbers higher than the last automatic order number. If **cobas IT 3000** cannot create a valid order number, the order is not created, and an error is written in the log files.

Configuring order result output

This section describes how to configure the **cobas IT 3000** to send results to the host.

In ASTM, the message consists of H, P, O, R and L records. In HL7, the message consists of the ORU^R01 event.

HL7 event	Interface event	Comment
ORU^R01	Result	Send in MSH-9. The 6th character of HL7 code is a zero.

Table A-24 HL7 event used for sending results to the host

To set up order result output to an host:

- Make sure you have the necessary information before starting.
- Check or configure **cobas IT 3000** it can send order results.
- Configure the **cobas IT 3000** administration options for sending results.
- Define the host client that handles communication with the host.
- Set up and start the LISA task (a Windows service) that the host client uses.
- Optionally, set up logging for the communication.

Before you start

Before you start, make sure you know, or have already configured in **cobas IT 3000**:

- The code the host needs for each analyte or profile (default HL7 OBR-4, or ASTM O-5.4)
- The internal codes your **cobas IT 3000** site uses for each:
 - Specimen
 - Analyte
- The XSLT stylesheet to use for your host connection for outgoing result messages. By default this is HL7_OUT.xml or ASTM_OUT.xml, available on GRIPS or from your Roche Diagnostics service personnel.
 - 👁 You can also copy them from Chapter 11 *The default XSLT transformations*.

If you use an edited stylesheet, discuss carefully what changes have been made with previous service personnel. Follow their advice, in addition to the set-up instructions here.

- The configuration file (.ini file) used on your site for the input LISA task. If you do not have this, you can create one from the instructions given later.

Configuring cobas IT 3000 parameters

First we have to set up the parameters for handling test results in **cobas IT 3000**. This involves:

- Creating the host client
- Setting up the general parameters
- Defining the analytes
- Setting up the result reports

Some or all of these parameters may already exist on your system. However, make sure they are correctly configured. Note that there is still configuration you need to do here, even if you have previously configured order input for this host.

Creating the host client

You need to define an “host client” inside **cobas IT 3000**. This holds the definition of the connection to the host, and contains an “Host ID” that other parts of the application use to identify that connection to that host. For now, just define the host client’s “Host ID”. We can complete the full details of the host client configuration later.

This may already be done on your system. However, make sure it is correctly configured.

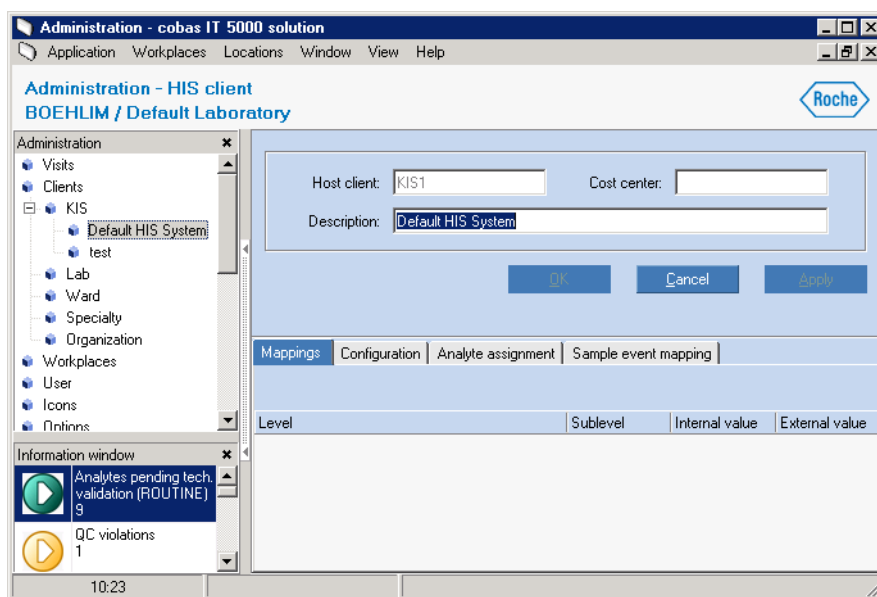


Figure A-19 Creating a host client and Host ID

► Create a host client

- 1 Navigate to **Workplaces > Administration > Locations > KIS**.
- 2 In the main window, from the context menu, select **Insert new Host location**. The **Host location** dialog opens.

Field or control	Enter	Comment
Host client	Short code to identify the connection.	Often called the “Host ID”. This is used in: <ul style="list-style-type: none"> the Location parameter in the configuration (.ini) file. the Output type or Host system field of the Report result configuration dialog. And many other places.
Description	The full name or description of the client.	
Cost center	Leave empty	

Table A-25 Entries in the **Host location** dialog

We can define the actual details of the output Host client connection later. For now, we need the “Host ID” field for the general configuration of **cobas IT 3000**.

- 3 Navigate to **Workplaces > Parameter > Organization**. In the main window, from the context menu, select **Insert new organization** or **Edit organization**. The **Organization** dialog opens.

Field or control	Enter	Comment
Organization code	Short code to identify the organization.	You cannot edit this later.
Host system	Select the host client that handles communication with this organization.	
(other fields)	Optional, but recommended.	Fill the details of the host or other host organization.

Table A-26 Entries in the **Organization** dialog

This defines the Host ID for the host client that handles connections to the organization. Next we need to make sure that the **cobas IT 3000** has sufficient general parameters set to handle incoming data.

Configuring cobas IT 3000 parameters

This section describes how to define the specialties, specimens and priorities of orders supported in this installation of **cobas IT 3000**.

► Configure cobas IT 3000 parameters

- 1 Navigate to **Parameter > Specialties**. From the context menu, select **Insert new specialty** or **Edit specialty**. In the **Specialty** dialog:

Field or control	Enter	Comment
Code	Short code	ID shown in the GUI. For example, SURG
Designation	Specialty	Name shown in the GUI. For example, Surgery

Table A-27 Entries to the **Specialty** dialog

Field or control	Enter	Comment
HIS code	Code sent by host	By default, sent in HL7 PV1-4. Not sent by default in ASTM.
Ward ID		Leave blank

Table A-27 Entries to the **Specialty** dialog

- 2 Navigate to **Parameter > Specimen**. From the context menu, select **Insert new specimen** or **Edit specimen**, to open the **Edit specimen** dialog.

Field or control	Enter	Comment
Code	Short code ID shown in the GUI.	For example, 01. This cannot be edited later.
Name (other fields)	Name shown in GUI.	For example, Serum Are optional (for order input)
Analytes list		You cannot assign analytes here. Assign them later in Parameter > Analytes / reference ranges . (Step 2)

Table A-28 Entries to the **Specimen** dialog

- 3 Navigate to **Parameter > Priorities**. From the context menu, select **Insert new priority** or **Edit priority**, to open the **Priority** dialog.

Field or control	Enter	Comment
Code	Short code ID, used internally, and sent by host.	For example, R. This cannot be edited later. By default, sent in ORC-7.
Name	Priority name shown in GUI.	For example, Routine or STAT
Priority	An integer code for priority. 1 has the highest priority.	
Color	The color the GUI will display for the order.	
Active	Select checkbox	
(other fields)		Are optional (for order input)

Table A-29 Entries to the **Priorities** dialog

Defining the analytes

You need to define the analytes supported in **cobas IT 3000**, if this has not yet been done.

► Define the analytes

- 1 Navigate to **Parameter > Analyte groups**. From the context menu, select **Insert new analyte group** or **Edit analyte group**, to open the **Edit analyte group** dialog.

Field or control	Enter	Comment
Code	Short code ID shown in the GUI.	For example, CC. This cannot be edited later.

Table A-30 Entries to the **Edit analyte group** dialog

Field or control	Enter	Comment
Name	Name shown in GUI.	For example, Clinical Chemistry.
Analytes list		You cannot assign analytes here. Assign them later in Parameter > Analytes / reference ranges . (Step 2)
(other fields)		Are optional (for order input)

Table A-30 Entries to the **Edit analyte group** dialog

- 2 Navigate to **Parameter > Analytes / reference ranges**. From the context menu, select **Insert new analyte** or **Edit analyte**, to open the **Analytes / reference ranges** dialog.

Field or control	Enter	Comment
Analyte no.	Internal ID number for analyte.	For example, 103. This cannot be edited later.
Analyte	Full name of analyte.	For example: Chloride.
Abbreviation	Internal abbreviation for analyte	For example: CL
Analyte group	Select from list.	Taken from Parameter > Analyte groups .
Specimen	Select from list.	Taken from Parameter > Specimen .
Module	Select LIS.	
Host code	Analyte code sent by host.	By default, sent in OBR-4.
Units	The units of measurement. (Such as mg/dl, mmol/L)	By default, sent in OBX-6.
Automatic release	Select checkbox	
Finalize after medical validation	Select checkbox	
Reference ranges list		See below.
(other fields)		Are optional

Table A-31 Entries to the **Analytes / reference ranges** dialog

Repeat this step for each analyte that the host uses.

- 3 Navigate to **Parameter > Analytes / reference ranges**. From the context menu, select **Insert new analyte** or **Edit analyte**, to open the **Analytes / reference ranges** dialog. Select the **Reference ranges** tab, and from the context menu, select **Insert new reference range** or **Edit reference range**. The **Reference range** dialog for that analyte opens.

Field or control	Enter	Comment
Name	Internal ID for the reference range.	
Normal range > Lower limit	A number	You can enter a minimum or maximum value instead.
Technical validation range > Upper limit	A number	You can enter a minimum or maximum value instead.

Table A-32 Entries to the **Reference ranges** dialog

Field or control	Enter	Comment
Normal range > Lower limit	A number	You can enter a minimum or maximum value instead.
Technical validation range > Upper limit	A number	You can enter a minimum or maximum value instead.
(other fields)		Are optional

Table A-32 Entries to the **Reference ranges** dialog

Repeat this step for each analyte that the host uses.

- 4 Navigate to **Parameter > Profiles**. From the context menu, select **Insert new Profile** or **Edit Profile**, to open the **Profiles** dialog.

Field or control	Enter	Comment
Profile code	Short code ID shown in the GUI.	For example, Electro. This cannot be edited later.
Profile name	Profile name shown in GUI.	For example, Electrolytes
Module	Select LIS.	
Host code	Code sent by host	By default, sent in OBR-4
Analyte list	Select analytes	Open Assign analytes dialog from the context menu in the lower list.
(other fields)		Are optional

Table A-33 Entries to the **Profiles** dialog

Defining the result reports

A test result has to be presented on a form. The form is put on a “result report”. Each result report is associated with an orderer. When a result is ready, **cobas IT 3000** chooses the result report for those analytes for that orderer. It sends the results to the host system associated with the result report.

► Define the result reports

- 1 Navigate to **Parameter > Forms**. Select form type **Individ. findings**, and then from the context menu, select **Insert new form** or **Edit form**, to open the **Form items** dialog.

Field or control	Enter	Comment
Form No.	Numerical ID.	
Form title	A descriptive title.	
(other fields)		Are optional

Table A-34 Entries to the **Forms** dialog

- 2 Specify which analyte groups will be in the report. In the **Form items** dialog, in the **Analyte group** list, from the context menu select **Insert new analyte group**, then in the **Forms analyte group** dialog, if necessary:
 - From the drop-down list, select an analyte group
 - In **Position**, enter a number. Lower numbers appear on the form before higher numbers.
- 3 Specify which analytes will be in the report. In the **Form items** dialog, select an analyte group in the left hand list. Then from the **Analyte** list, from the context

menu select **Insert new form position** or **Edit form position**. Then in the **Forms analyte position** dialog, if necessary:

- From the drop-down list, select an analyte
- In **Position**, enter a number. Lower numbers appear on the form before higher numbers.

- 4 Navigate to **Parameter > Result report configuration**. From the context menu, select **Insert new configuration** or **Edit configuration**, to open the **Result report configuration** dialog.

Field or control	Enter	Comment
Config. No.	Short code ID.	This cannot be edited later.
Name	A human-readable name for the configuration	This name identifies the configuration in the Orderer dialog.
Output type	The Host ID of your host client.	Either this or “ Host system ” must match the Host client ID.
Form type	Select “ Individ. findings ”	
Form	The form configured in step 1.	
File	Must be “none”	Unless “ Host system ” is set, in which case you may optionally define a file.
Result selection > Not sent to orderer	Selected	
Mark results > Sent to orderer	Selected	
Decimal symbol	Choose a decimal point or comma.	
Host system	The Host ID of your host client.	Either this or “ Output type ” must match the Host client ID.
(other fields)		Are optional (for order input)

Table A-35 Entries to the **Result report** dialog

- 5 Navigate to **Parameter > Orderer**. From the context menu, select **Insert new orderer** or **Edit orderer**, to open the **Orderer** dialog.

Field or control	Enter	Comment
Code	Short code ID shown in the GUI.	For example, WA1. This cannot be edited later.
Orderer	Orderer’s name shown in GUI.	For example, Ward 1
Organization	Your organization.	Taken from Parameter > Organization .
Host code	Code sent by host	By default, sent in PV1-3
Report type	The report entered in step 4.	
Auto. print type	The report entered in step 4.	

Table A-36 Entries to the **Orderer** dialog

Field or control	Enter	Comment
Emergency rep.	The report entered in step 4.	
Ward ID	The code sent by the host	
Ward communication	Selected	Activates communication via host
Specialty list	At least one specialty.	Open Specialty dialog from the context menu in the lower list.
(other fields)		Are optional

Table A-36 Entries to the **Orderer** dialog

Configuring cobas IT 3000 administration options

This section describes how to configure **cobas IT 3000** administration options to send test results to this host. These settings might already be configured on your system, but check them carefully.

This configuration sends results to the host after they have been medically validated. Some installations send this at technical validation. We can consider reconfiguration later.

► Configure cobas IT 3000 administration options

- 1 Navigate to **Workplaces > Administration > Trigger Administration**. Set the option “**Send Trigger for medically validated results**” to **ENABLED**. The option may also be referred to as BEFUND_EMB_VAL_RESULTATE.

When a result has been medically validated, this trigger fires. At this point in time, **cobas IT 3000** sends results to the host.

- 2 Navigate to **Administration > Options > Log**.

Create an option TRACE_LEVEL if it does not already exist. (Context menu, **Insert new entry. Option group** dialog.)

Field or control	Enter	Comment
Item	TRACE_LEVEL	
Content	40	Set this option to 0 (OFF) before going into a production environment.
Group	log	
Description	Trace level for LISDB messages. 0 = OFF, 10 = ERROR, 20 = WARNING, 30 = INFO, 40 = DEBUG.	
(other fields)	(Are optional for order input)	

Table A-37 Entries to the **Administration > Options > Log > TRACE_LEVEL** dialog

- 3 Navigate to **Administration > User**. Expand the tree to show [operator's user name] > **Rights > KIS > [host ID]**.

In the main window, select the row marked **Read patient data (SEL_PAT)**, and from the context menu, select **Assign**.

Also assign these rights in the same way:

- **Insert patient** (INS_PAT)
- **Delete patient** (DEL_PAT)
- **Update patient** (UPD_PAT)

Defining the Host client

Earlier we defined the Host ID and the organization it referred to. We left the full definition of the Host client connection to later. Now is the time to fill this in. This involves:

- Checking the Host ID for the client is defined.
- Creating the XSLT configuration.
- Mapping the analyte codes.
- Creating and adding a configuration (.ini) file.
- Defining the LISA task, and adding it to the list of LISA tasks.
- Setting up the interface logging. This is optional, but recommended for test environments.

Check your host client

Earlier, we defined an host client to handle the connection to the host. This is defined in **Workplaces > Administration > Locations > KIS**, and referenced in **Workplaces > Parameters > Organization**, in the field **Host system**. Check these are set.

 For details of we created this, see *Creating the host client* on page A-48.

The **cobas IT 3000** handles messages to or from that host according to the settings in the Host client configuration.

Creating the XSLT configuration

Navigate to **Workplaces > Administration > Locations > KIS > [Your host client] > Configuration**. If there is no XSLT configuration, with the **Root name** OrderResults, create one.

► To create the input XSLT configurations:

- 1 Make sure that you have the file HL7_OUT.xml or ASTM_OUT.xml to hand, and it is open and ready for you to cut and paste from.
- 2 Navigate to **Workplaces > Administration > Locations > KIS > [Your host client] > Configuration**.
- 3 From the context menu, select **Create new configuration**. To view or edit a pre-existing configuration, double-click it.

The **Maintain configuration** dialog opens.

- 4 Enter/select the following:

Field or control	Content	Comment
Configuration	Enter a name	Example: HL7_OUT
Active	Select	

Table A-38 Values for the upper part of the **Maintain configuration** dialog.

Field or control	Content	Comment
Root name	OrderResults	The root name of the XML created by cobas IT 3000 for order results. 👁 For a full list of root names, see <i>Identifying which XSLT to use</i> on page A-12.
Queue	A unique ID string. This must be different from other configurations in this Host client.	This is the Queue for Oracle Alert messages from the database. This must match the QUEUE parameter in the configuration (.ini) file.
Direction	OUT	Direction of data transfer

Table A-38 Values for the upper part of the **Maintain configuration** dialog.

Target settings:

Field or control	Content	Comment
Application	Hospital Information System	
Queue	A unique ID string. This must be different from other configurations in this Host client.	This is the Queue for Oracle Alert messages from the database. This must match the QUEUE parameter in the configuration (.ini) file.
Send after (sec.)	0	
Retry after (sec.)	10	
Wait for Ack. (sec.)	5	
Location	Select your host client	
Direction	OUT	Direction of data transfer
Number of attempts	5	
Priority	Medium	

Table A-39 Values for the **Target settings** part of the **Maintain configuration** dialog.

- 5 Copy and paste (Ctrl+C and Ctrl+V) the content of the file HL7_IN.xml into the text field at the bottom of the dialog. Make sure your cursor is at the very beginning of the text field before you paste the XSL content!
- 6 Choose **OK**.
- 7 If you receive the error message "**Unimplemented or unreasonable conversion requested**", your file contains too many characters. In this case, delete all empty spaces before the tags:
 - Place the cursor in the text field.
 - Press Ctrl+R.
 - In **Text to find**, type two empty spaces.
 - Make sure the field **Replace with** is completely empty.
 - Click **OK**. The XSLT is saved and the dialog closes.

This tells **cobas IT 3000** how to process the message from the client. When a message arrives, **cobas IT 3000** looks at:

- client name or **Location**, in this case the host client we defined earlier.
- The **Application**, which must be **HIS** for host connections.
- The **Direction**, in this case **OUT** (from **cobas IT 3000** to host).
- The **Queue**, which is the Oracle queue, and distinguishes this configuration from other configurations with the same details.

From this information, **cobas IT 3000** decides which configuration to use when parsing the message.

Mapping analyte codes

You must map the analyte codes used by the host to the codes used by the **cobas IT 3000**.



Note

Before you can map analyte codes, you must create the analytes in **Parameter > analytes / reference ranges**.

► To map analytes:

- 1 Navigate to **Administration > Locations > KIS > [Your host client] > Mappings**
- 2 From the context menu, select **insert new mapping**.
The **Create new analyte mapping dialog** appears.
- 3 In **Description**, enter the analyte abbreviation.
- 4 From the **Internal Analyte** list, select the relevant analyte.
- 5 In **External analyte (In)**, enter the analyte code received from the host.
- 6 In **External analyte (Out)**, enter the analyte code that should be sent to the host (usually this is the same as in **External analyte (in)**).
- 7 Click **<OK>**.
- 8 Repeat steps 2-7 for all analytes (tests, or substances tested for) that can be requested from the host.

This tells **cobas IT 3000** the meaning of the codes to send to the host.



WARNING

The internal and external mapping must be unique

The mapping from an external code to an internal code is a one-to-one relationship. Each incoming external code must map to a unique internal code. Each internal code must map to a unique outgoing external code.

Adding INI file to database

The **cobas IT 3000** uses configuration (often called .ini or initialization) files to define parameters and DLLs used by the communication server when setting up the host connection.

Make sure you have a configuration (.ini) file prepared. If you do not, create one using this example and edit it match your site, according to the following instructions.

*Example result configuration
(.ini) file*

Make sure you have a configuration (.ini) file prepared. If you do not, create one using this example and edit it match your site, according to the following instructions.

```

/*****
* PatientResultHL7.ini
*****/

[GENERAL]
Applications=HOST

[HOST]
Connections=RESULT,LIS

[HOST-RESULT]
DllName          = LISDB
Destination      = HOST-LIS
AutoDelete       = 1
Application       = HIS
Location         = KIS1
Driver           = OCIOA10
Database         = CIT5K-SERVER01/LIS
NLS_NUMERIC      = .,
Queue            = OUT
Encoding         = ISO-8859-1
WriteThreadTimeout = 90000
ReadThreadTimeout  = 90000
ReConnectThreadTimeout = 90000

[HOST-LIS]
DllName          = LIS
PortConfig       = 7,,c:\data\out\out%05d.hl7,10
BaseProtocol     = 203,4000,0,0
Parser           = MSXML
Encoding         = ISO-8859-1
#Destination     =
MessageType      = HL7
TraceFile        = C:\data\log\resultshl7.log

```

What this does

The first section of the .ini file

```

[GENERAL]
Applications=HOST

```

says that this file defines an application that it names HOST.

The next section

```

[HOST]
Connections=RESULT,LIS

```

says that the HOST application uses two modules, that it names RESULT and LIS.

The next section:

```

[HOST-RESULT]
DllName          = LISDB
...
Destination      = HOST-LIS

```

says that the HOST-RESULT module is a LISDB.dll module. The **cobas IT 3000** can send an order result in XML format to a LISDB.dll module. The LISDB.dll module transform this according to an XSLT file into an XML file that describes the structure of an HL7 or ASTM message. It sends the result to the module named in the Destination parameter.

The next section:

```
[HOST-LIS]
DllName      = LIS
```

says that the HOST-LIS module is a LIS.dll module. If an LIS.dll module receives an XML file that describes the structure of an HL7 or ASTM message, it creates the actual HL7 or ASTM, and sends it to a location specified in the further parameters, described below.

► Editing the .ini file

- 1 Use the example above to start with.
- 2 If necessary, edit the header comment to the name of the file, for example PatientResultASTM.ini.
- 3 Edit the values for the LIS module.

Parameter	Value	Comment
PortConfig	The full filepath to your output files. If necessary, correct the file type tag to .astm or .dat.	You must create this folder if it does not exist. The out%05d value creates a rolling 5 digit number for the files.
MessageType	HL7 or ASTM	Can handle one kind of message or the other, but not both.
TraceFile	Complete filepath.	For debugging, LIS.dll writes all HL7 or ASTM messages to this file.
(Other parameters)	Can be left as the example for now.	

Table A-40 Parameters for the LIS module

If necessary, you can configure a TCP/IP port or other kind of connection, later. To do so, change the PortConfig parameter, and check the values of the BaseProtocol and Protocols parameters.

👁 For details, see *Using a TCP/IP connection* on page A-68.

- 4 Edit the values for the LISDB module.

Parameter	Value	Comment
Application	HIS	Tells cobas IT 3000 to process this as an host connection.
Location	Host ID of your host client	Workplaces > Administration > Locations >[your host client]. Take the value in the Host client field.

Table A-41 Parameters for the LISDB module

Parameter	Value	Comment
Queue	Defines the value of the Oracle queue. Distinguishes the configuration from others with the same Application and Location.	This must match the value in Workplaces > Administration > Locations > [your host client] > Configuration > [select the configuration] > Maintain Configuration dialog > Queue .
Database	The login details of your cobas IT 3000 database.	This is the same as the value for Database in the jboss.ini file in Workplaces > Administration > LISA Administration > Configuration .
(Other parameters)	Can be left as the example for now.	

Table A-41 Parameters for the LISDB module

You can now add the configuration (.ini) file to **cobas IT 3000**.

► Adding the configuration (.ini) file

- 1 Navigate to **Workplaces > Administration > LISA Administration > Configuration**.
- 2 From the context menu, select **Insert new configuration**. The **Create** dialog opens.
- 3 In **ID** and **Name**, enter the name of your file, for example, "PatientResultHL7.ini" or "PatientResultASTM.ini".
- 4 In **Description**, explain that this file is used to receive order messages from the host. (optional)
- 5 Copy and paste the content of the configuration (.ini) into the text field at the bottom of the dialog.
- 6 Click **Save**, and close the dialog. You have set up the configuration file.

Creating a LISA task

LISA tasks launch the parts of the application specified in the INI files. You must create a task which activates the host interface and sends output messages (results) to the host. Creating a LISA task involves two steps:

- Adding a task to the LISA configuration file
- Starting the task

Adding a new LISA task

If there is no LISA task that starts your host interface, create one in **Workplaces > Administration > LISA Administration > Tasks**.

► **Creating or editing the LISA task definition**

- 1 Go to **Workplaces > Administration > LISA Administration > Tasks**. If there is a definition for your host interface, double click on it. Otherwise to create a new task, from the context menu select **Add new task**.
- 2 In the dialog, edit the following as necessary.

Attribute	Value	Comment
LISA server	The location of the database where the LISA task definitions are stored. Usually localhost.	This cannot be edited later.
External ID	Enter an ID for the task.	This is displayed in the LISAMGR application. This cannot be edited later.
Task name	Enter a name that you can use to identify this task.	This is displayed in LISA tasks.
Command	This is a command passed to the operating system. Enter the filepath of the communication server program, host.exe. Add the parameters In this example: %CIT5K_CS3K5K%\host.exe / I:PatientResultsH77. ini /M:0xFFFFFFFF	/I: tells host.exe to load the named .ini file. Enter the name of the .ini file, in Workplaces > LISA Administration > Configuration . /M: 0xFFFFFFFF sets the maximum log level. In production, reduce this to 0x100000.
Directory	The location of the bin directory of the cobas IT 3000 installation.	
Autostart	Select this to start the task automatically when cobas IT 3000 starts.	
Inherit environment	Select this.	Inherits the environment variables and settings from cobas IT 3000 .

Table A-42 Creating or editing a LISA task

Starting the LISA order results task

Before you can start sending orders from the host, you must start the host task.

► **To start the LISA host task**

- 1 Navigate to **Workplaces > Administration > LISA Administration > Task**.
- 2 Right-click your mouse onto the host task in the table and select **Task > Start**.
The task is started and the system starts checking for incoming messages (requests) from the host.

When the communication server receives a message, it processes it according to the rules defined in the configuration (.ini) file, and then sends it to the host, or in our case, writes it to the output directory.

Setting up host interface logging (optional)

You can set detailed debugging for the communication channel, using the log4j functionality. Make sure that **Administration > Options > Log > TRACE_LEVEL** is set.

► To create a debugging task

- 1 Navigate to **Workplaces > Administration > LISA Administration > Tasks**.
- 2 From the context menu, select **Insert new configuration**.
- 3 The **Create** dialog opens.
- 4 In **ID** and **Name**, enter “PatientResultsHL7.xml”, or the name you gave to your order configuration, with the tag “.xml”.
- 5 In **Description**, explain that this file defines the log4j debugging of “PatientResultsHL7.ini”, or whatever you called the file.
- 6 Copy and paste the contents of the file below into the text field at the bottom of the dialog. Edit the file, so that it shows the log file you wish to write to.

```
<?xml version='1.0' ?>
<log4j:configuration debug='false' xmlns:log4j='http://jakarta.apache.org/log4j/'>
<appender name='DEFAULT' class='org.apache.log4j.RollingFileAppender'>
  <param name='File' value='c:/data/log/PatientResultsHL7.log' />
  <param name='MaxFileSize' value='50MB' />
  <param name='MaxBackupIndex' value='20' />
  <layout class='org.apache.log4j.PatternLayout'>
    <param name='ConversionPattern' value='Trace %d{ISO8601} [%t] %-5p %c %x - %m\n' />
  </layout>
</appender>

<root>
  <priority value='DEBUG' />
  <appender-ref ref='DEFAULT' />
</root>
</log4j:configuration>
```

Figure A-20 The log4j file

Click **Save**, and close the dialog. Log information will be written to the specified file.

Sending example order results to the host

After you have configured your order results host interface, test it with some sample messages.

Before you start Make sure your HL7 or ASTM order result host task is running.

Make sure that you have some orders in **cobas IT 3000**. To do this you can either:

- Send the orders from the host as described above.
 - 👁 For details of how to send order requests from the host, see *Setting up external order input* on page A-24.
- Navigate to **Workplaces > Routine > Order input**, and input some orders manually.

► Send results to the host

- 1 Navigate to **Workplaces > Routine > Day list**.
- 2 If you cannot see the orders, you wish to send, from the context menu, choose **Search criteria**. Use the **Selection** dialog box to search for orders.
- 3 Select an order to send to the host. The details of the order appear in the lower window.

The order will be sent to the host associated with the orderer.
- 4 If any analyte does not have results, and you cannot generate results on the instrument or from an instrument simulator, enter results manually.
 - To enter results manually, select an analyte, and press <Return>. The **Result entry** dialog appears.
 - Enter the result value in **Result**, and press <Return>. The dialog closes.
 - Repeat this step for all analytes in the order.
- 5 Technically validate each result. Select the analyte, and from the context menu, select **Technical validation**.
- 6 Navigate to **Workplaces > Routine > Medical Validation**. From the context menu, select **Search criteria**, and use the search dialog to find the technically validated result.
- 7 Select the order in the upper window, and from the context menu, select **Medical validation of order**. The order appears in the output directory defined in the configuration .ini file.

An HL7 result should look as follows:

```
MSH|^~\&|PACMAN|PACMAN-LAB|PSM|PSM-LAB|20110308174423||ORU^R01|8877|P|2.4|||ER|NE||8859/1|
PID|1||444444||Coray^Ruth||19301213|W|
PV1|1|||||||||||||||||||||||||||||||||||||
ORC|1|8763524|||20110308105123|
OBR|1|635246||102^102||20110308174357|||||||||||||||||||||||||||||||||||||N|
OBX|1||102^102^||4.0^ok!|mmol/L|3.7 - 5.4^3.7^5.4|||F||20110308174357|^|BOEHLIM^08-MAR-
11^BOEHLIM^08-MAR-11|||
OBR|2|635246||104^104||20110308174401|||||||||||||||||||||||||||||||||||||N|
OBX|2||104^104^||2.400^No normal range predefined!|mmol/L| - ^|||F||20110308174401|^|BOEHLIM^08-
MAR-11^BOEHLIM^08-MAR-11|||
```

Figure A-21 Example HL7 result report

An ASTM result should look as follows:

```
H|\^&|||PACMAN|||PSM|P|20110308180248|
P|1|8764000108|||Coray^Ruth||19301213|W|
O|1|8764000108|^^^^^|ALL||20110308140453|F|
R|1|^^^102|4.0|mmol/L|||F|||20110308180210|||N|
R|2|^^^104|2.500|mmol/L|||F|||20110308180213|||N|
L|1|N|
```

Figure A-22 Example ASTM result report

Troubleshooting

For debugging, make sure that **Administration > Options > Log > TRACE_LEVEL** is set to 40. Reset this to 0 after troubleshooting.

You can see log information of the communication in **Workplaces > Administration > Communication messages**. The successful message reads, going across the columns: **Pass - HIS - [Host ID] - [Queue]- OrderResults - OUT - OrderResults OK**. Double click on the message, and the **Information** tab reads **OrderResults OK**, and the lower panel has tabs showing:

- **Old content:** The first XML content that **cobas IT 3000** created when it validated the order.
- **XML content:** The final XML content that **cobas IT 3000** used to create the HL7 or ASTM message.

This communication log often has useful information in case something has gone wrong. There are further suggestions here.

The order does not appear in the output directory.	Check: <ul style="list-style-type: none">• The correct LISA task is running.• The output directory matches the value in the .ini file in PortConfig.• The file type of the message in the directory matches the file type specified the .ini file in PortConfig.• Check the permissions on the output directory: can cobas IT 3000 write files there?
The LISA task won't start.	Check: <ul style="list-style-type: none">• The LISA task is correctly named and defined in the list of LISA tasks. Look at the entry in Administration > LISA administration > Nodes.• The database is correctly defined in the LISA task. Look at the entry in Administration > LISA administration > Nodes.
In Administration > Communication messages , message reads "Fail".	Make sure the Queue parameter in the configuration (.ini) file matches the Queue field in the Host client definition in Administration > Locations > [Host client] Configuration > [config] .
Double click the message, and the Information tab reads "Not supported root name <...>".	

Table A-43 Order result output, troubleshooting suggestions.

The order is validated, but has not appeared in the output directory. The log file specified in the log4j config has a message similar to:

```
ERROR BCNormalHostThread - HIS Exception
<HostAppLayerError> on handle of <HL7>
```

or:

```
ERROR BCNormalHostThread - HIS Exception
<HostAppLayerError> on handle of <ASTM>
```

In **Administration > Communication messages**, the message reads “Pending” or “Fail”, but there is no log information in the tabs **Error logs** and **Trace logs**.

The order does not appear in the output directory, nor in **Administration > Communication messages**.

The order does not appear in the output directory, nor in **Administration > Communication messages**.

But it is marked as “Sent to host” in **Routine > View results**.

In **Administration > Communication messages**, the message reads “Fail”, and the **Message name** is “ASTM”.

Double click the message, and the **Information** tab reads “Not supported root name <ASTM>”.

In **Administration > Communication messages**, the message reads “Fail”, and the **Message name** is “HL7”.

Double click the message, and the **Information** tab reads “Not supported root name <HL7>”.

Check that the **Destination** parameter is correctly defined in the configuration (.ini) file.

Remember that the whole name of the destination must be specified, with a hyphen: APP-MODULE, for example:

```
Destination = HOST-LIS
```

In **Administration > Options > Log** set TRACE_LEVEL = 40.

Check:

- The correct LISA task is running.
- The trigger is enabled in **Administration > Trigger administration**.
- In **Parameters > Orderer**, the result’s orderer has a Result report assigned to it, in the field **Result report**.
- Your order had a sample ID assigned to it.
- The host task in **Administration > Locations > [host client] > Configuration > [output configuration]** has the **Root name** “OrderResults” (without a space).

Check **Parameters > Result report configuration**. Either the field **Output type** or **Host system** must be set to the Host ID of your host client.

If using HL7:

In the configuration (.ini) file make sure MessageType = HL7.

If using ASTM:

Make sure the **Queue** parameter in the configuration (.ini) file matches the **Queue** field in the Host client definition in **Administration > Locations > [Host client] > Configuration > [config]**.

If using ASTM:

In the configuration (.ini) file make sure MessageType = ASTM.

If using HL7:

Make sure the **Queue** parameter in the configuration (.ini) file matches the **Queue** field in the Host client definition in **Administration > Locations > [Host client] Configuration > [config]**.

Table A-43 Order result output, troubleshooting suggestions.

Send results on technical validation

You can configure analytes to send their results to the host in two circumstances.

- After *technical validation*, when the technicians have confirmed that the results are valid.
- After *medical validation*. This occurs after technical validation is completed, when the medical staff also confirm that the result is valid.

The configuration of result export messages described previously sent the results on medical validation. This section describes how to configure analytes so that they send their results on technical validation.

► To send results on technical validation

- 1 Navigate to **Workplaces > Parameter > Analytes / reference ranges**. Double-click on the analyte you wish to configure. The **Analyte / reference ranges** dialog opens.

Field or control	Enter	Comment
Automatic release	Select checkbox	
Finalize after medical validation	Deselect checkbox	
(other fields)		Keep unchanged.

Table A-44 Entries to the **Analytes / reference ranges** dialog

👁 For fuller details of setting analyte and reference ranges, see the online help, or *Defining the analytes* on page A-50.

- 2 Navigate to **Workplaces > Administration > Trigger Administration**. Set the option “**Send Trigger for finalized results**” to **ENABLED**. The option may also be referred to as BEFUND_EMB_FIN_RESULTATE.

When a result has been technically validated, this trigger fires. At this time, **cobas IT 3000** sends the results to the host.

Grouping results

By default **cobas IT 3000** groups the results by specimen, in the result message to the host. Configure this in **Workplaces > Administration > Locations > KIS > [host client] > Options**. Results can be grouped by:

- Specimen: one message per sample.
- Analyte group: one message per analyte group.
- Profile: one message per profile.
- Order number, then by analyte group: one message per order.
- Order number, then by profile: one message per order.

Using a TCP/IP connection

This section describes the configurations that must be set in **cobas IT 3000**, and the operating system to implement host communications over TCP/IP. To implement this you need to:

- Configure the system files “hosts” and “services”.
- Configure the port and protocols in the configuration (.ini) file.
 - Define the port and host with the `PortConfig` parameter.
 - Define the framing protocol, such as the starting and ending characters, with the `BaseProtocol` parameter.
 - If necessary, define the flow protocol, such as the ACK conditions, with the `Protocols` parameter.

Before you start, consult with your system administrator or host developer, to find out:

- what hosts and port you are communicating on
- what protocols you are using, including the framing protocol
- what flow and acknowledgement protocols you are using.

The following sections give guidance, but you may want to consult your system administrator and the host developer for information about your system.

System files

The operating system (MS Windows) needs to know about the hosts and the ports that **cobas IT 3000** is using. The hosts are defined in the “hosts” file, and the ports are defined in the “services” file. By default in an MS Windows system, these files are stored in:

```
c:\windows\system32\drivers\etc
```

hosts The file “hosts” contains IP addresses and assigned host names. The format of an entry is:

```
<ip address> <host name> [#<comment>]
```

Example:

```
172.18.18.18 cobasIT3000 # cobas p242 server
```

Make sure that the IP address and the host name of your host have are unique in the file.

services The services file contains well known ports defined by IANA and additionally your local entries. The format of an entry is:

```
<service name> <port number>/<protocol> [aliases...] [#<comment>]
```

Example:

```
cobasIT3k 12345/tcp #cobas IT 3000 service
```

Make sure that the service name and the port number of your service are unique in the file.

Setting up the HL7 connection

Edit your host connection in **Workplaces > Administration > LISA administration > Tasks**, for both the order input and the result output configurations.

Edit the definition of the LIS module as follows.

Parameter	Value	Comment
PortConfig	Use as a server:	Info is an ID string used in the log file.
	1, Info, Service	
	Use as a client:	Host is name or IP address.
	32, Info, Host, Service	
BaseProtocol	402,4096	Service is name or port number.
Protocols	101	The second parameter is buffer size
(other fields)		Sets up flow control using ASK / NAK.
		Keep unchanged.

Table A-45 Editing the LIS module in the HL7 configuration file to support TCP/IP

Setting up the ASTM connection

Edit your host connection in **Workplaces > Administration > LISA administration > Tasks**, for both the order input and the result output configurations.

Edit the definition of the LIS module as follows.

Parameter	Value	Comment
PortConfig	Use as a server:	Info is an ID string used in the log file.
	1, Info, Service	
	Use as a client:	Host is name or IP address.
	32, Info, Host, Service	
BaseProtocol	303,8192	Service is name or port number.
Protocols	102	The second parameter is buffer size
(other fields)		Sets up flow control using ASK / NAK.
		Keep unchanged.

Table A-46 Editing the LIS module in the ASTM configuration file to support TCP/IP

Network port configuration

This section describes the definitions for network ports. These are defined in the configuration file for your host connection, in **Workplaces > Administration > LISA administration > Tasks**. The host machine and port are defined with the parameter PortConfig, normally part of the definition of the LIS module. For example:

```
[HOST-LIS]
DllName      = LIS
```

```
#Portconfig = 32,Info,Host,Service
PortConfig   = 2,47_client,COBAS-COM5,47_server
BaseProtocol = 402,4096
Protocols    = 201
Parser       = MSXML
Encoding     = ISO-8859-1
#Destination =
MessageType  = ASTM
TraceFile    = C:\data\log\resultsastm.log
```

This section describes the TCP/IP settings for the `PortConfig` parameter.

TCP/IP server

A TCP/IP connection provides a Socket Server listening on a single port. It can handle only one connection at the time:

Syntax: `Portconfig = 1,Info,Service`

<i>Parameters</i>	Info	String that will used in the trace to identify the port
	Service	TCP/IP service name on which the socket server will listen Has to be defined in the system file “services”.

TCP/IP client

Two TCP/IP clients are available: one creates a TCP/IP connection on demand. The other one keeps the network connection open, even if currently no data is send over the network.

2	TCP/IP Client with a connection on demand
32	TCP/IP Client with a permanent connection

Syntax `Portconfig = 2,Info,Host,Service` (Connection on demand)
`Portconfig = 32,Info,Host,Service` (Permanent connection)

<i>Parameters</i>	Info	String that will used in the trace to identify the port
	Host	Remote Server to which the client shall connect. Host name must be used. If DNS server is not configured, then the system file “hosts” is used.
	Service	TCP/IP service name on which the socket server will listen Has to be defined in the system file services

Data framing and flow protocols

This section describes the low level data framing and flow protocols. These are configured in the configuration file (the .ini file) for the host connection, in the definition of the LIS module. The parameters that describe protocols are:

- `BaseProtocol`. This describes the framing protocol, such as the starting and ending characters.
- `Protocols`. This describes the flow protocol, such as the ACK conditions.

HL7 configuration

The HL7 TCP/IP interface uses VT/FS dataframes, which uses the following protocols.

<i>Syntax</i>	<code>BaseProtocol = 402,BufferSize</code>	
<i>Parameters</i>	<code>BufferSize</code>	The buffer size passed to the protocol
<i>Example</i>	<code>Portconfig = 32,HL7Patient,pluto,patient</code> <code>BaseProtocol = 402,4096</code>	

HL7 flow control

HL7 communication supports flow control by sending ACK (Acknowledgement) or NAK (No Acknowledgement / Negative Acknowledgement).

<i>Syntax</i>	<code>Protocols = 101</code>
<i>Example</i>	<code>Portconfig = 1,ASTMPatient,patient</code> <code>BaseProtocol = 402,8192</code> <code>Protocols = 101</code>

ASTM configuration and flow control

This section describes the low level protocol (framing and flow control) according to ASTM 1381, implemented in two protocol layers.

<i>Syntax</i>	<code>BaseProtocol = 303,BufferSize</code> <code>Protocols = 102</code>	
<i>Parameters</i>	<code>BufferSize</code>	The buffer size passed to the protocol
<i>Example</i>	<code>Portconfig = 1,ASTMPatient,patient</code> <code>BaseProtocol = 303,8192</code> <code>Protocols = 102</code>	

Buffer Size

This section give guidelines on setting the buffer size for communication protocols.

In the definitions of the base protocols, the second parameter is the buffer size. The value to give this depends on the length of the messages being processed.

<i>General guideline</i>	<p>As a general guideline, the buffer size should be:</p> $\text{BufferSize} = \text{max number of characters in message} + \text{security margin}$ <p>All messages that exceed the buffer size will not be processed. Therefore a generous security margin should be chosen. Too high a value should also be avoided, as it might lead to performance issues.</p>
<i>Maximum value</i>	The value should not be set higher than 15,000, as this could cause problems with the stability of the host interface.

Further protocols supported

There are many other protocols that **cobas IT 3000** supports, including socket connections, semaphore files, and more. For details, please consult your Roche Diagnostics service personnel or the ***cobas IT 3000 Service Manual***.

HL7 interface

B

4	<i>HL7 protocol</i>	B-3
5	<i>Description of the HL7 Interface (LISDB)</i>	B-7

HL7 protocol

The low-level definition of HL7

This chapter presents the lower layers of the HL-7 protocol, as used by **cobas IT 3000**.

In this chapter	Chapter 4
HL7 protocol lower level	B-5
Physical communication	B-5
Minimal Layer Protocol	B-6

HL7 protocol lower level

This chapter gives a low-level description of the HL7 protocol used to communicate with **cobas IT 3000**.

HL7 or “Health Level 7” is one the most widely-used protocols in the healthcare business. Nevertheless, the standard is so large that **cobas IT 3000** uses only a subset.

The HL7 Standard currently addresses the interfaces among various systems that send or receive patient admissions/registration, discharge or transfer (ADT) data, queries, resource and patient scheduling, orders, results, clinical observations, billing, master file update information, medical records, scheduling, patient referral, and patient care. It does not try to assume a particular architecture with respect to the placement of data within applications but is designed to support a central patient care system as well as a more distributed environment where data resides in departmental systems.

In **cobas IT 3000**, HL7 v2.4 is implemented.

Physical communication

The standard HL7 refers to the highest level of the Open System Interconnection (OSI) model of the International Standards Organization (ISO). The HL7 Standard is primarily focused on the issues that occur within the seventh, or application, level. These are the definitions of the data to be exchanged, the timing of the exchanges, and the communication of certain application-specific errors between the applications. This chapter gives some recommendations for how to use HL7 with **cobas IT 3000**.

Minimal Layer Protocol

Data framing is done using the Minimal Layer Protocol (MLP) defined in the HL7 standard, (sometimes referred to as MLLP, Minimal Lower Layer Protocol).

HL7 messages are enclosed by special characters to form a block. The format is as follows:

<start_block>data<end_block><CR>	
<start_block>	Start Block character (1 byte) ASCII <VT>, i.e., <0x0B>. This should not be confused with the ASCII characters SOH or STX.
data	Data (variable number of bytes) This is the HL7 data content of the block. The data can contain any displayable ASCII characters and the carriage return character, <CR>.
<end_block>	End Block character (1 byte) ASCII <FS>, i.e., <0x1C>. This should not be confused with the ASCII characters ETX or EOT.
<CR>	Carriage Return (1 byte) The ASCII carriage return character, i.e., <0x0D>.

The values used are <VT> for the start block and <FS> for the end block.



Figure B-1 The HL7 data framed according to the HL7 standard.

Description of the HL7 Interface (LISDB)

This part of the document shows the different message types available to communicate between a Laboratory Information System (LIS) and **cobas IT 3000** solution using the HL7 protocol.

In this chapter

Chapter 5

HL7 communications	B-9
Minimal Layer Protocol	B-9
Sending data to cobas IT 3000	B-10
Configuring the interface with parameters	B-10
Sending patient data	B-10
Supported HL7 fields	B-10
Admitting a new patient	B-12
Transferring a patient	B-13
Updating patient information	B-13
Sending order requests	B-14
Message structure	B-14
Field mappings	B-15
New Order	B-19
Update order	B-19
Replacement order	B-20
Delete a sample	B-20
Delete an order	B-20
Handling different actions in the order	B-20
Update delete level	B-21
Automatic creation of orderer	B-22
The fields available in the ORDERER element	B-23
Receiving data from cobas IT 3000	B-25
Receiving observation results	B-25
Message structure	B-25
Configuring the interface with parameters	B-26
Field mappings	B-27
Receiving an acknowledgement message from an order request message	B-32

Message structure B-32

Configuring the interface with parameters B-32

Field mappings B-33

Receiving quality control results B-35

 Message structure B-35

 Configuring the interface with parameters B-35

 Field mappings B-36

 Values not mapped by default B-39

Sample Event message B-41

 Message structure B-41

 Configuring the interface with parameters B-41

 Field mappings B-42

HL7 communications

Roche Diagnostics’ **cobas IT 3000** solution can manage a group of instruments for a laboratory information system (LIS). The **cobas IT 3000** solution’s module connects as a client to the LIS server, and communicates over TCP/IP. Thus the **cobas IT 3000** solution provides a single interface through which the LIS can communicate with a group of instruments.

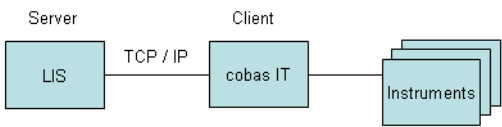


Figure B-2 cobas IT 3000 solution managing a group of instruments.

The LIS system can send and receive messages in HL7 format to and from the **cobas IT 3000** solution. Roche Diagnostics’ **cobas IT 3000** solution handles all further communication with its group of instruments.

Minimal Layer Protocol

Data framing is done using the Minimal Layer Protocol (MLP) defined in the HL7 standard, (sometimes referred to as MLLP, Minimal Lower Layer Protocol).

HL7 messages are enclosed by special characters to form a block. The format is as follows:

<start_block>data<end_block><CR>

<start_block>	Start Block character (1 byte)
k>	ASCII <VT>, i.e., <0x0B>. This should not be confused with the ASCII characters SOH or STX.
data	Data (variable number of bytes) This is the HL7 data content of the block. The data can contain any displayable ASCII characters and the carriage return character, <CR>.
<end_block>	End Block character (1 byte)
>	ASCII <FS>, i.e., <0x1C>. This should not be confused with the ASCII characters ETX or EOT.
<CR>	Carriage Return (1 byte) The ASCII carriage return character, i.e., <0x0D>.

The values used are <VT> for the start block and <FS> for the end block.



Figure B-3 The HL7 data framed according to the HL7 standard.

Sending data to cobas IT 3000

This section describes messages sent from the host (LIS) to **cobas IT 3000**, when using the default stylesheets. For each message type, you will find answers to the following questions:

- What is the message used for?
- What segments must the message consist of?
 - [] = optional segments
 - {} = segments which can be repeated
- Which fields does **cobas IT 3000** use?
- Which fields are required and which fields are optional?
- How can you configure the interface with parameters?

Configuring the interface with parameters

You can configure the host interface by using the parameters in **Workplaces > Administration > Clients > [your client] > Configuration**. The default stylesheet provides the following for messages sent to **cobas IT 3000**:

Parameter	Type	Default value	Other possible values	Instruction to cobas IT 3000:
acks	Boolean	true	false	Respond to application and accept acknowledgement instructions.
patid	Integer	3	4, 5	Which field in the PID segment contains the patient ID.
sampleid	Integer	2	3, 4	Which field in the OBR segment contains the sample ID.
ordernr	Integer	2	3, 4	Which field in the ORC segment contains the order number

Table B-1 Parameters used for incoming messages.

Sending patient data

Patient messages send data for the following events from a host (LIS) to **cobas IT 3000**:

- Admitting a new patient
- Transferring a patient
- Discharging a patient
- Updating patient information

All HL7 events (ADT^A01, ADT^A02, etc.) can be mapped to those four events.

Supported HL7 fields

The following list contains a full list of patient-related HL7 fields supported by **cobas IT 3000**. Further down, you will find details about the required segments for each of the supported tasks/events.

HL7 Segment	Pos. in HL7	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
MSH	1	ST			The delimiter, set to the bar symbol().	
MSH	2	ST			The other encoding characters. Set to ^~\&.	
MSH	3	ST	180	Optional	Sending application	<MSH SENDER> and <PID HIS>
MSH	9	ID	7	Required	Message type (9.1) and event (9.2), e.g. ADT^A01, ADT^A02, etc.	<MSH MESSAGE_TYPE>
MSH	10	ST	20	Required under conditions.	Message control ID (unique number to identify the message, e.g. the time the message is generated: YYYYMMDDHHMMSSxxxx) Required if MSH.15 below is AL, SU, ER.	<MSH CONTROL_ID>
MSH	15	ID	2	Required if "Use Ack info"=true	Defines conditions for sending accept acknowledgment messages. Requiring accept messages only makes sense with socket-based messaging, but not with file-based messaging. <ul style="list-style-type: none"> AL = Always (Default if blank) SU = Success ER = Error NE = Never 	<MSH ACCEPT_TYPE>
MSH	16	ID	2	Optional	Defines conditions for sending application acknowledgment messages. <ul style="list-style-type: none"> AL = Always SU = Success ER = Error NE = Never 	<MSH ACK_TYPE>
PID	3	ST	20	Required	Patient ID. If the HL7 message does not contain a patient ID, cobas IT 3000 creates its own ID	<PID PATID>
PID	5.1	ST	48 in total	Optional	Patient name: Surname	<PID LASTNAME>
PID	5.2	ST	48 in total	Optional	Patient name: First name	<PID FIRSTNAME>
PID	7	TS	14	Optional	Date of birth (format: YYYYMMDD)	<PID DOB>
PID	8	IS	1	Optional	Sex: F=Female; M=Male; U=Unknown. Default is U.	<PID SEX>
PV1	2	IS	1	Optional	Admission status. User-defined class. Suggested values: E=Emergency, I=Inpatient, O=Outpatient, P=Preadmit, R=Recurring patient	<PV1 CLASS>
PV1	3	IS	80	Optional	Assigned patient location: Ward, point of care	<PV1 WARD>
PV1	4	IS	80	Optional	Specialty	<PV1 SPECIALTY>
PV1	19	ST	20	Optional	Visit no. Unique number to identify the visit	<PV1 VISITNR>

Sending data to cobas IT 3000

HL7 Segment	Pos. in HL7	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
PV1	44	TS	26	Required if used.	Admission date/time (format: YYYYMMDDHHMMSS). Required in an admit patient message.	<PV1 ADMITDATE>
PV1	45	TS	26	Required if used.	Discharge date/time (format: YYYYMMDDHHMMSS). Required in a discharge patient message.	<PV1 DISCHARGEDATE>
DG1	3.1	ST		Optional	Diagnosis code	<DG1 ICDCODE>
DG1	3.2	ST		Optional	Text description of the diagnosis code	<DG1 TEXT>
DG1	5	TS	14	Optional	Date/time the diagnosis was determined (format: YYYYMMDDHHMMSS).	<DG1 DATE>
DG1	6	ID	1	Optional	Diagnosis type. Suggested values: <ul style="list-style-type: none"> • A = Admitting • W = Working • F = Final 	<DG1 TYPE>
IN1	3	ST		Optional	Insurance company code string	<IN1 IN1>

(a) Data type abbreviations: ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for HL7-defined tables

Admitting a new patient

An *ADT^A01* message should be sent to create a patient and/or a corresponding visit in **cobas IT 3000**.

Upon receipt of the message, **cobas IT 3000** checks if the patient exists already.

- If the patient already exists, **cobas IT 3000** updates any patient details contained in the message and, if there is a new visit no. (case no.), creates a new visit for the patient.
- If the patient does not yet exist, **cobas IT 3000** creates a new patient and creates the first visit for the patient.

Message structure

The message must contain the following segments:

Segment	Segment name	Description
MSH	Message header	General information about message, including the message type (mandatory)
PID	Patient identification	Patient ID and other patient data (mandatory)
PV1	Patient visit	Information about the patient visit (mandatory to admit a patient; optional to transmit patient details only)

Table B-2 Structure of a message to admit a patient

Sample message

Use case:

Mrs. Martina Higgins, born 10 January 1963, was admitted to ward E7 on 18 April 2007 at 09:30. Her stay at the hospital was assigned visit no. 223277.


```
MSH|^~\&|LIS|||||ADT^A01|200704180930150003|||||ER|ER|
PID|||HM10011963F||Higgins^Martina||19630110|F|
PV1||I|E7|||||||||||||223277|||||||||||||200704180930150003
```

Figure B-4 Sample message.

Transferring a patient

An *ADT^A02* message should be sent as a result of a patient changing his or her physical location (ward, room, speciality).

Upon receipt of the message, **cobas IT 3000** changes the location of the patient (HL7 fields PV1-3.1, PV1-3.2 and PV1-3.4) and updates any other fields if the data differs from the details in the system.

Message structure The message must contain the following segments:

Segment	Segment name	Description
MSH	Message header	General information about message, including the message type (mandatory)
PID	Patient identification	Patient ID and other patient data (mandatory)
PV1	Patient visit	Information about the patient visit (mandatory)

Table B-3 Structure of a message to transfer a patient

Sample message

Use case:
Mrs. Martina Higgins, born 10 January 1963, was transferred to ward S15.

```
MSH|^~\&|LIS|||||ADT^A02|200704191012340001|||||ER|ER|
PID|||HM10011963F||Higgins^Martina^^Mrs.||19630110|F|
PV1||I|S15|||||||||||||223277|||||||||||||200704180930150003
```

Figure B-5 Sample message.

Updating patient information

An *ADT^A08* should be sent to update any details about the patient OR the current visit.

Upon receipt of the message, **cobas IT 3000** updates any patient or visit details contained in the message.

Message structure To update patient information, the message must contain the following segments:

Segment	Segment name	Description
MSH	Message header	General information about message, including the message type (mandatory)
PID	Patient identification	Patient ID and other patient data (mandatory)
[PV1]	Patient visit	Information about the patient visit (mandatory to change details about the visit)
DG1	Diagnosis	Information about the patient's diagnosis (optional)
IN1	Insurance	Information about the patient's insurance (optional)

Table B-4 Structure of a message to update patient information

Use case:
This message updates the data for Mark Pearson, born 09 March 1971, giving the code number for his insurance company.

```
MSH|^~\&|LIS|||||ADT^A08|200702030855030001|||1||ER|ER|
PID|||PM09031971M||Pearson^Mark^^Mr.||19710309|M|||45 Great Eastern Road^^Glasgow^^G1A 1BA|
IN1|||0509|
```

Figure B-6

Sample message.

Sending order requests

Order requests transfer the following data from a host (LIS) to **cobas IT 3000**:

- New orders (test requests)
- Changes to an existing order
- Deletion of an existing order
- Patient data (with every order request, patient data included in the order is updated and patients who do not yet exist are created)

**WARNING**

Danger of patient data inconsistency received from host.

Do not use this application as a main repository for patient data.

Make sure that the host (LIS/HIS) vendor always sends a unique patient ID for each patient to **cobas IT 3000**.

Any patient demographic changes coming from the host will be updated, no matter if there are results already validated and sent to the host for that specific patient. This may lead to potential wrong validation flags for the affected patient results.

Avoid this kind of change in patient demographic data.

**CAUTION**

Patient ID must be ASCII 7 string

The patient ID must be an ASCII 7 string. Whitespace, extended ASCII or unicode is not supported.

Message structure

An Order request contains the following segments:

Segment	Segment name	Description
MSH	Message header	General information about message, including the message type (mandatory)
PID	Patient identification	ID used to match order with patient data sent previously (mandatory)
[PV1]	Patient visit	Information about the patient visit (mandatory if visit number used, otherwise optional)
[DG1]	Diagnosis	Information about the patient's diagnosis (optional)
[IN1]	Insurance	Information about the patient's insurance (optional)
ORC	Common order data	General information about order (mandatory, more than 1 ORC segment per message possible)
{[NTE]}	Comments on order	Comments about order (optional, but more than 1 NTE segment per ORC is possible)
{		

Table B-5 Structure of an order request

Segment	Segment name	Description
OBR	Observation request	Details about required test (mandatory)
[OBX]	Observation result	Clinical information needed to interpret the result (optional)
}		

Table B-5 Structure of an order request


Field mappings

This list contains all supported HL7 fields. By default, any other fields that can be contained in an HL7 order message are ignored.

HL7 Segment	Pos. in HL7	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
MSH	1	ST	1	Required	The delimiter, set to the bar symbol().	
MSH	2	ST	4	Required	The other encoding characters. Set to ^~\&.	
MSH	3	ST	180	Optional	Sending application.	<MSH SENDER> and <PID HIS>
MSH	9	ID	7	Required	Message type (9.1) and event (9.2), e.g. ORM^O01 or OML^O21	<MSH MESSAGE_TYPE>
MSH	10	ST	20	Required	Message control ID (unique number to identify the message, e.g. the time the message is generated: YYMMDDHHMMSSxxxx). Required if MSH.15 below is AL, SU, ER.	<MSH CONTROL_ID>
MSH	15	ID	2	Required if "Use Ack info"=true	Defines conditions for sending accept acknowledgment messages. Requiring accept messages only makes sense with socket-based messaging, but not with file-based messaging. <ul style="list-style-type: none"> AL = Always (Default if blank) SU = Success ER = Error NE = Never 	<MSH ACCEPT_TYPE>
MSH	16	ID	2	Required if "Use Ack info"=true	Defines conditions for sending application acknowledgment messages. <ul style="list-style-type: none"> AL = Always (Default if blank) SU = Success ER = Error NE = Never 	<MSH ACK_TYPE>
PID	3	ST	20	Required	Patient ID. If the HL7 message does not contain a patient ID, cobas IT 3000 creates its own ID	<PID PATID>
PID	5.1	ST	48 in total	Optional	Patient name: Surname	<PID LASTNAME>
PID	5.2	ST	48 in total	Optional	Patient name: First name	<PID FIRSTNAME>
PID	7	TS	14	Optional	Date of birth (format: YYYYMMDD)	<PID DOB>
PID	8	IS	1	Optional	Sex: F=Female; W=Female; M=Male; U=Unknown. Default is U.	<PID SEX>

Roche Diagnostics

Sending data to cobas IT 3000

HL7 Segment	Pos. in HL7	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
PV1	3	IS	80 in total	Optional	Assigned patient location: Ward, point of care; order enterer's location	<PV1 WARD>
PV1	4	IS	80 in total	Optional	Specialty	<PV1 SPECIALTY>
PV1	19	ST	20	Required if used.	Visit no. Unique number to identify the visit. May be empty if there is no associated visit.	<PV1 VISITNR>
PV1	44	TS	26	Optional	Admission date/time (format: YYYYMMDDHHMMSS).	<PV1 ADMITDATE>
PV1	45	TS	26	Optional	Discharge date/time (format: YYYYMMDDHHMMSS).	<PV1 DISCHARGEDATE>
DG1	3.1	ST		Optional	Diagnosis code	<DG1 ICDCODE>
DG1	3.2	ST		Optional	Text description of the diagnosis code	<DG1 TEXT>
DG1	5	TS	14	Optional	Date/time the diagnosis was determined (format: YYYYMMDDHHMMSS).	<DG1 DATE>
DG1	6	ID	1	Optional	Diagnosis type. Suggested values: <ul style="list-style-type: none"> • A = Admitting • W = Working • F = Final 	<DG1 TYPE>
IN1	3	ST		Optional	Insurance company code string	<IN1 IN1>
ORC	1	ID	2	Required	Order control ID. There are 3 possible field contents: <ul style="list-style-type: none"> • NW=New order; • XO=Change existing order; • CA=Delete existing sample (to delete an order, all samples in the order must be deleted)  For more details, see <i>Handling different actions in the order</i> on page B-20.	<ORC CONTROL>
ORC	2	ID	15	Optional	Order ID or Specimen ID The order ID is a unique number that identifies the order. By default this is in field 2 ("Placer Order Number") of the ORC record, but it is possible to reassign it to field 3 ("Filler Order Number") or to field 4 ("Placer Group Number"). See the parameters in cobas IT 3000 Administration > Client > Configuration.	<ORC ORDERNR>

HL7 Segment	Pos. in HL7	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
ORC	5	ID	2	Optional	<p>Order status ID. Supported values are:</p> <ul style="list-style-type: none"> • A = Add a test • CA = Sample was deleted (to cancel an order, all samples in the order must be deleted) • SC = In process, scheduled • R = Order has been replaced • RP = Order has been replaced • UP = Update order. This recreates the order with specified tests, plus any previous tests that fulfil the conditions specified in the option HIS_UPDATE_DELETE_LEVEL. <p>👁 For details of HIS_UPDATE_DELETE_LEVEL, see <i>Update delete level</i> on page B-21.</p> <p>This value is used in conjunction with the Order Control ID, passed in ORC-1, and the Action Code, passed in OBR-11, to instruct cobas IT 3000 how to handle the order.</p> <p>👁 For more details, see <i>Handling different actions in the order</i> on page B-20.</p>	<ORC STATUS>
ORC	7	ST		Optional	<p>Priority</p> <ul style="list-style-type: none"> • R = Routine • S = STAT <p>Note that this is in the first component of this field. This differs from the HL7 standard, which puts this value in component 6 of this field.</p>	<ORC PRIORITY>
ORC	9.1	TS	26	Required	Date/time of order entry	<ORC ORDERDATE>
ORC	9.2	TS	26	Optional	The format of the date/time of order entry, passed in field ORC-9.1	<ORC ORDERDATEFMT>
ORC	10	ST	120	Optional	The orderer. The person who entered the order, or other source of the order. (Required if no default orderer is set in cobas IT 3000 , and the results are going to be returned to the host.)	<ORC ORDERER>
OBR	2	NM	22	Required	<p>By default, the Sample ID.</p> <p>Sample ID created by LIS. This is a unique number that identifies the sample. By default this is in field 2 ("Placer Order Number") of the OBR record, but it is possible to reassign it to field 3 ("Filler Order Number") or to field 4 ("Universal Service Identifier"). See the parameters in cobas IT 3000 Administration > Client > Configuration.</p>	<OBR SAMPLEID>
OBR	4	ST	200	Required	Universal Service ID, i.e. Required test code	<OBR TESTCODE>

HL7 Segment	Pos. in HL7	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
OBR	7.1	TS	26	Optional	Date and time the specimen was taken. Defaults to system time.	<ORC WITHDRAWDATE>
OBR	7.2	TS	26	Optional	The format of the date and time the specimen was taken, passed in field OBR-7.1.	<ORC WITHDRAWDATEFMT>
OBR	11	ID	2	Required	The specimen action code. Supported values are: <ul style="list-style-type: none"> • A = Add ordered test to existing specimen • G = Generated order, reflex order • R = Revised order <p>👁 For more details, see <i>Handling different actions in the order</i> on page B-20.</p>	<OBR ACTION>

(a) Data type abbreviations: ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for HL7-defined tables

You cannot send orders containing results

Although the default templates contain fields for orders containing results, this is not supported in the current release. This may be supported in future releases. If you require this functionality, please contact Roche Diagnostics Global Customer support.

Values not mapped by default

There are further values that the host can send to **cobas IT 3000** in an order request, but are not transferred by the default XSL templates. If you wish to use these values, you will need to edit the XSL templates to support them.

👁 For details of how the XSL templates control the data passed between **cobas IT 3000** and the host, see *Reading the ASTM or HL7 messages* on page A-10.

HL7 Segment	Pos. in HL7	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
-	-	ST		Optional	It is possible to use an alphanumeric alternative for the order number or sample number. In this case, the host should send the alphanumeric code identifying the order. Up to 30 characters are supported. This is suggested as a value for field ORC-2, component 1 "Placer Order Number", but in some circumstances, ORC-3 (Filler Order Number), or ORC-4 (Placer Group Number) might be an alternative.	<ORC HOST_ORDER_ID>

HL7 Segment	Pos. in HL7	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
-	-	ST		Optional	The host's code identifying the physician dealing with the patient. This is suggested as a value for field PV1-7, component 1 "Attending physician code ID", but in some circumstances, PV1-8 (Referring doctor), PV1-9 (Consulting doctor) might be an alternative.	<ORC PHYSICIAN_CODE>
-	-	ST		Optional	The name of the physician. This is suggested as a value for field PV1-7, component 2 "Attending physician name". Note that currently, cobas IT 3000 does not support dividing the name into last name, first name, title, etc.	<ORC PHYSICIAN_NAME>
-	-	IS		Optional	A boolean value Y or N, telling cobas IT 3000 to add the physician's details to its database, if the details are not already there. The default setting is Y. To find the physician's preexisting details, cobas IT 3000 searches for a match for the physician's host code. If the code is not supplied, cobas IT 3000 searches by name for a match.	<ORC PHYSICIAN_CREATE>

(a) Data type abbreviations: ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for HL7-defined tables

New Order

This example shows a message ordering three new tests, coded 114, 115, 116, are ordered for Karl Müller, born 6th June 1970.

```
MSH|^~\&|LIS|||||OML^O21|200703251640330001||||ER|ER|
PID|||4711||Müller^Karl||19700606|M|
PV1|||S2^^NEU|||||||||||6|
ORC|NW|8763401||||R||20070325160023|
OBR||8763401||114||20090912210000|||A
OBR||8763401||115||20090912210000|||A
OBR||8763401||116||20090912210000|||A
```

Figure B-7 Sample message ordering new tests.

Update order

This example shows an update order (XO in the Control Code: ORC-1. UP in the Status Code: ORC-5. A in the OBR record). A in the Action Code: OBR-11

```
MSH|^~\&|HIS|||||ORM^O01|200703251640330001||||ER|ER|
PID|||4711||Müller^Karl||19700606|M|
PV1|||S2^^NEU|||||||||||6|
ORC|XO|8763401||UP||R||20070325160023|
OBR||8763401||114||20070325153000|||A
OBR||8763401||115||20070325153000|||A
OBR||8763401||116||20070325153000|||A
```

Figure B-8 Sample message updating an order

Replacement order

This example shows a replacement order. (XO in the Control Code: ORC-1. R in the Action Code: OBR-11.)

```
MSH|^~\&|HIS|||ORM^O01|200703251640330001|||ER|ER|
PID||4711|Müller^Karl||19700606|M|
PV1||S2^^NEU||||||6|
ORC|XO|8763401|||R|20070325160023||^Randall^Tom^^Dr|
OBR||8763401|VKBB||20070325153000||R
```

Figure B-9 Sample message updating an order

Delete a sample

This example shows a message deleting a sample. (CA in the Control Code: ORC-1. R in the Action Code: OBR-11.)

```
MSH|^~\&|HIS|||ORM^O01|200703251640330001|||ER|ER|
PID||4711|Müller^Karl||19700606|M|
PV1||S2^^NEU||||||6|
ORC|CA|8763401|||R|20070325160023||^Randall^Tom^^Dr|
OBR||8763401|ELE||20070325153000||R
```

Figure B-10 Sample message deleting a sample

Delete an order

This example shows a message deleting an order. (CA in the Control Code: ORC-1.)

```
MSH|^~\&|HIS|||ORM^O01|201110101414141001|||ER|ER|
PID||4711|Müller^Karl^^Mr.||19700606|M|
ORC|CA|8763401|||R|20111010141414||^Randall^Tom^^Dr|
OBR||101||20111010141414|
OBR||102||20111010141414|
OBR||103||20111010141414|
```

Figure B-11 Sample message deleting an order

Handling different actions in the order

This table explains how to instruct **cobas IT 3000** to perform actions on the order, using the fields in the Order record.

	ORC Control (ORC-1)	ORC Status (ORC-5)	OBR Action code (OBR-11)
Add test	NW		A
Delete test	XO	A	R
Delete order (if no sample ID is sent, complete order will be deleted; if sample ID is sent, only the sample will be deleted)	CA		
Rebuild new sample (delete all tests for the sample and add the new ones)	XO		R
Delete sample (delete all tests for the sample and delete the sample itself)	XO	CA	R
Delete sample (as above)	CA		R
Delete order	CA		
Change in PID	XO	SC	R
Rerun / reflex	XO	RP	G
Update order	XO	UP	A

To cancel an order, all samples in the order must be deleted.

Update delete level

When an update order message is sent, the order is recreated with the new tests specified. The order's previous tests are deleted, except those that meet the criteria specified in the parameter HIS_UPDATE_DELETE_LEVEL. This parameter is set in **cobas IT 3000** in Administration > Options > MODUL > HIS_UPDATE_DELETE_LEVEL.

The parameter HIS_UPDATE_DELETE_LEVEL is a bitwise binary flag. It consists of an 8-digit binary number, each digit of which can be either 1 or 0. Each digit sets a different flag, as detailed in the following chart.

Do not delete if test:	Binary flag	Code
(Never delete)	10000000	DEL_LVL_NEVER
Has been transferred	01000000	DEL_LVL_TRANSFERRED
Has been scanned	00100000	DEL_LVL_SCANNED
Has been put on work list	00010000	DEL_LVL_WORK_LIST
Has been seen	00001000	DEL_LVL_SEEN
Has been processed by SDI	00000100	DEL_LVL_DONE_BY_SDI
Has a result	00000010	DEL_LVL_RESULT
Has a released result	00000001	DEL_LVL_RELEASED_RESULT

As a binary flag, these values can be combined. For example, the setting 00011011 tells **cobas IT 3000** not to delete any pre-existing tests that meet one or more of the following criteria:

- tests that have been added to the worklist (00010000)
- tests that have been seen by **cobas IT 3000** and sent to the instrument (00001000)
- tests that have a result (00000010)
- tests that have a released result (00000001)

The default value of HIS_UPDATE_DELETE_LEVEL is 00011011.

Automatic creation of orderer

If using the default templates, you must pre-program **cobas IT 3000** with the orderers that will send orders. If you send an orderer which **cobas IT 3000** does not know of, it cannot return test results. The default templates read the orderer information from the HL7 segment ORC, field ORC-10, and assign it to ORC/@ORDERER.

However, you can configure **cobas IT 3000** to automatically create a new orderer, whenever it receives an order from a new source. In this case, you must configure **cobas IT 3000** to read the orderer information from the ORDERER element. The HL7 Order Request message must be transformed into an XML document similar to the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<OrderRequest>
  <MSH SENDER="TEST"/>
  <PID PATID="TH" LASTNAME="HARRY" FIRSTNAME="TEST" DOB="19010101" SEX="M">
    <ORC ORDERNR_ORG="1453376" ORDERNR="453376" SAMPLEID="1453376" SPECIMEN="" PRIORITY="R"
STATUS="" CONTROL="NW">
      <ORDERER DEFAULT_AUFGCD="?" HOSTCODE="WARD1" NAME="WARD1" WARDID="WARD1" WARD="J"/>
      <OBR TESTCODE="NA" ACTION="A"/>
      <OBR TESTCODE="K" ACTION="A"/>
      <OBR TESTCODE="CREAT" ACTION="A"/>
      <OBR TESTCODE="MG" ACTION="A"/>
    </ORC>
  </PID>
</OrderRequest>
```

Figure B-12 Example Order Request message for automatic creation of orderer.

To implement this, you must edit the XSLT stylesheets for the Order Request.

👁 For details of how to find and edit the XSLT stylesheets, see *Making extensive reconfigurations* on page A-15.



How to edit XSLT

To edit the XSLT stylesheets, you should learn at least some basic XSLT and XPath.

You can find tutorials and reference information on XSLT and XPath on the internet, for example at <http://www.w3schools.com>. Please note that Roche Diagnostics is not responsible for the content of this or any other external website.

► To implement automatic creation of orderer in ASTM

- 1 Confirm in what HL7 field the host sends the orderer (or sender) information. Typically this will be in ORC-10, but some sites send it in another field.
- 2 Navigate to **Administration > Client (or Location) > [Name of Location] > Configuration**, and double click the template for order requests, by default **OrderRequest**. A dialog box opens, containing the Order Request XSLT.
- 3 Copy and paste all the content of the dialog box to your preferred XML editor (such as Stylus Studio or Altova XML Spy), or to your preferred text editor (such as Notepad or Notepad++). Make a backup copy, and then edit the file in your chosen editor.
- 4 Find and delete the section in the template that assigns a value to the attribute "ORDERER".

- 5 Find the definition of the last attribute of the ORC element. The ORC element starts with the tag:

```
<xsl:element name="ORC">
```

It ends with

```
</xsl:element>
```

Inside this element there are several attributes, defined with tags:

```
<xsl:attribute ...> ... </xsl:attribute>
```

There is also a sub-element, OBR, defined with the tags:

```
<xsl:element name="OBR"> ... </xsl:element>
```

Put your new commands between the last `</xsl:attribute>` tag and the `<xsl:element name="OBR">` that starts the OBR element.

- 6 Enter the XSLT commands to assign the orderer information to the ORDERER element. For example, if the orderer information is sent in HL7 field ORC-10, add, as the first sub-element in the ORC element:

```
<xsl:for-each select="FIELD[@NO='10']/DATA/COMP[@NO='1']">
  <xsl:element name="ORDERER">
    <xsl:attribute name="DEFAULT_AUFGCD"?></xsl:attribute>
    <xsl:attribute name="HOSTCODE"><xsl:value-of select="@TEXT"/></xsl:attribute>
    <xsl:attribute name="NAME"><xsl:value-of select="@TEXT"/></xsl:attribute>
    <xsl:attribute name="WARDID"><xsl:value-of select="@TEXT"/></xsl:attribute>
    <xsl:attribute name="WARD">J</xsl:attribute>
  </xsl:element>
</xsl:for-each>
```

Figure B-13 Example XSLT to read automatic orderer creation data from ASTM O-17.

If the orderer information is sent in another HL7 field, edit this example to meet your specific needs.

- 7 Save your changes in the editor, paste the contents of the edited file back into **cobas IT 3000**, and save your changes in **cobas IT 3000**.
- 8 Make a note of the default orderer you defined in the XSLT, in the `DEFAULT_AUFGCD` element ("?" in this example). Make sure this orderer is defined in **Workplaces > Parameter > Orderer**. Every orderer field not defined in the orderer element will be taken from the default orderer.
- 9 Test your host interface thoroughly before implementing it live.

The fields available in the ORDERER element

The ORDERER element is not used by the default **cobas IT 3000** templates. By editing the templates, the following fields are available. For more details on these fields see the online help for the dialog in **Workplaces > Parameter > Orderer > Insert new orderer**.

Automatic creation of orderer

HL7 Segment	Pos. in HL7	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
-	-	ST		Optional	The name of the default orderer. The default orderer must exist in Workplaces > Parameter > Orderer . Every orderer field not defined in the orderer element will be taken from the default orderer.	<ORDERER DEFAULT_AUFGCD>
-	-	ST	5	Required	The code used in the host for the orderer.	<ORDERER HOSTCODE>
-	-	ST		Required	The name of the orderer.	<ORDERER NAME>
-	-	ST		Optional	Text description of the orderer, to help a user with identification in reports.	<ORDERER REMARK>
-	-	ST		Optional	The cost center associated with the orderer.	<ORDERER COSTCENTER>
-	-	ST		Optional	The code that determines the order in which orderers appear on reports.	<ORDERER SORT>
-	-	ST		Optional		<ORDERER TITLE1>
-	-	ST		Optional		<ORDERER TITLE2>
-	-	ST		Optional		<ORDERER ZIP>
-	-	ST		Optional		<ORDERER CITY>
-	-	ST		Optional		<ORDERER BILLINGCODE>
-	-	ST		Optional	A pseudo-boolean value. Optional, but must be set to “J” in production environments, or else results are not returned.	<ORDERER WARD>
-	-	ST		Optional	(This functionality is not currently supported.)	<ORDERER TOUR>
-	-	ST		Optional	The orderer group defined in cobas IT 3000 , in which to place the new orderer.	<ORDERER ORDERERGROUP>
-	-	ST		Optional	The organization to associate the orderer with in cobas IT 3000	<ORDERER ORGANIZATION>
-	-	ST		Optional	The code to give to the orderer in cobas IT 3000 .	<ORDERER ABBREVIATION>
-	-	ST		Optional		<ORDERER PHONE>
-	-	ST		Optional		<ORDERER FAX>
-	-	ST		Required	The ward ID for the new orderer, used in ward communication.	<ORDERER WARDID>

(a) Data type abbreviations: ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for HL7-defined tables

Receiving data from cobas IT 3000

This section describes messages sent from **cobas IT 3000** to the host (LIS). For each message type, you will find an answer to the following questions:

- What is the message used for?
- What segments does the message consist of?
 - [] = optional segments
 - {} = segments which can be repeated
- Which fields does **cobas IT 3000** create for outgoing HL7 messages?
- What parameters can you use to configure the outgoing message?

Receiving observation results

Observation results transfer test results to the application (LIS) which ordered the test(s). The following data is sent from **cobas IT 3000** to the host (LIS):

- Results for requested tests

Message structure

An **Observation result** contains the following segments:

Segment	Segment name	Description
MSH	Message header	General information about message, including the message type
PID	Patient identification	Patient information such as ID, name and first name, DOB and sex
PV1	Patient visit	Information about the patient visit
{		
ORC	Common order data	General information about order such as the order creator, the creation date and the order status
OBR	Observation request	Details about required test
OBX	Observation result	Test result. This includes the actual result, the unit, the reference ranges, etc.
[[NTE]]	Comments on result	Comments about the result and its interpretation
}		

Table B-6 Structure of an observation result

Configuring the interface with parameters

You can configure the host interface by using the parameters in **Workplaces > Administration > Clients > [your client] > Configuration**. The default stylesheet provides the following for messages with observation result messages from **cobas IT 3000**:

Parameter	Type	Default value	Other possible values	Instruction to cobas IT 3000:
patid_pos	Integer	3	4, 5	Send the patient ID in this field in the PID segment.
posSID	Boolean	true	false	True: put the sample ID in OBR-2. False: put the order number in OBR-2.
SENDING_APPLICATION	String		(any)	The value to write in MSH-3, for the “sending application”. MSH-3. If set, this value overrides any value configured in cobas IT 3000 .
SENDING_FACILITY	String		(any)	The value to write in MSH-4, for the “sending facility “. If set, this value overrides any value configured in cobas IT 3000 .
RECEIVING_APPLICATION	String		(any)	The value to write in MSH-5, for the “receiving application”. If set, this value overrides any value configured in cobas IT 3000 .
RECEIVING_FACILITY	String		(any)	The value to write in MSH-6, for the “receiving facility”. If set, this value overrides any value configured in cobas IT 3000 .
ACCEPT_TYPE	String	AL		The value to write in MSH-15, as the accept acknowledgement type: <ul style="list-style-type: none"> • AL = Always (Default if blank) • SU = Success • ER = Error • NE = Never Using this only makes sense with socket-based messaging. If set, this value overrides any value configured in cobas IT 3000 .
ACK_TYPE	String	AL		The value to write in MSH-16, as the application acknowledgement type: <ul style="list-style-type: none"> • AL = Always (Default if blank) • SU = Success • ER = Error • NE = Never If set, this value overrides any value configured in cobas IT 3000 .

Table B-7 Parameters used for incoming messages.

Field mappings

This list contains the HL7 fields generated by **cobas IT 3000** and provides a short description for each field.

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
MSH	1	ST		The delimiter, set to the bar symbol().	
MSH	2	ST		The other encoding characters. Set to ^~\&.	
MSH	3	ST		Sending application. It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH SENDING_APPLICATION>
MSH	4	ST		Sending facility (laboratory). It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH SENDING_FACILITY>
MSH	5	ST		Receiving application. It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH RECEIVING_APPLICATION>
MSH	6	ST		Receiving facility (LIS). It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH RECEIVING_FACILITY>
MSH	7	TS		Date/time message is generated	<MSH DATE>
MSH	9.1	ID		Message Type, in this case "ORU"	<MSH MESSAGE_TYPE>
MSH	9.2	ID		Message Event, in this case "R01"	<MSH EVENT_TYPE>
MSH	10	ID		Message control ID. The cobas IT 3000 echoes the value the host sent to in this field. This sends an acknowledgement to the HIS. Required if MSH-15 below is AL, SU, ER.	<MSH CONTROL_ID>
MSH	11	ID		Processing ID. This tells the systems to use the HL7 processing rules. In this case, this is set to P = Production.	<MSH PROCESSING_ID>
MSH	12	ID		Version ID. This shows the version of HL7 used in the message.	<MSH VERSION>

Receiving data from cobas IT 3000

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
MSH	15	ID	2	<p>Defines conditions for sending accept acknowledgment messages. Requiring accept messages only makes sense with socket-based messaging, but not with file-based messaging.</p> <ul style="list-style-type: none"> • <i>AL</i> = Always (Default if blank) • <i>SU</i> = Success • <i>ER</i> = Error • <i>NE</i> = Never <p>If using the default templates, it is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.</p>	<MSH ACCEPT_TYPE>
MSH	16	ID	2	<p>Defines conditions for sending application acknowledgment messages</p> <ul style="list-style-type: none"> • <i>AL</i> = Always (Default if blank) • <i>SU</i> = Success • <i>ER</i> = Error • <i>NE</i> = Never <p>If using the default templates, it is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.</p>	<MSH ACK_TYPE>
MSH	18	ID		Defines the character set used in the message. (utf-8, ISO-8859-1, etc.)	<MSH ENCODING>
PID	1	NM		Set ID. Sequence number of segment (numbers multiple segments of the same type).	<PID SET>
PID	3	ST		Patient ID	<PID PATID>
PID	5.1	ST		Patient last name	<PID LASTNAME>
PID	5.2	ST		Patient first name	<PID FIRSTNAME>
PID	7	TS		Date of birth	<PID DOB>
PID	8	IS		Sex	<PID SEX>
PV1	1	NM		Set ID. Sequence number of segment (numbers multiple segments of the same type).	<PV1 SET>
PV1	19	NM		Visit no.	<PV1 VISITNR>
PV1	44	TS	26	Admission date/time (format: YYYYMMDDHHMMSS)	<PV1 ADMITDATE>
PV1	45	TS	26	Discharge date/time (format: YYYYMMDDHHMMSS)	<PV1 DISCHARGEDATE>
ORC	1	NM		Set ID. Sequence number of segment (numbers multiple segments of the same type)	<ORC SET>

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
ORC	2	NM		Order number	<ORC ORDERNR>
ORC	9	TS		Date/time the order was requested (format: YYYYMMDDHHMMSS)	<ORC REQUESTDATE>
OBR	1	NM		Set ID. Sequence number of segment (numbers multiple segments of the same type)	<OBR SET>
OBR	2	NM		By default, the Sample ID. The user can set this field to show the Order Number. To do so, set the <code>posSID</code> parameter in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration to “false”.	<OBR SAMPLEID> or <ORC ORDERNR>
OBR	4.1	IS		External test code (i.e code used by LIS)	<OBX ID>
OBR	4.2	IS		Internal test code (i.e. code used by cobas IT 3000)	<OBX ANALYTNR>
OBR	7	TS		Date/time test result was received by cobas IT 3000 , considered to be the time the test was carried out.	<OBX COMPLETEDATE>
OBR	49	IS	1	The confidentiality of the test result. This is set for the test, by a parameter in cobas IT 3000 : <ul style="list-style-type: none"> C = confidential test N = normal test 	<OBX CONFIDENTIAL>
OBX	1	NM		Set ID. Sequence number of segment (numbers multiple segments of the same type)	<OBX SET>
OBX	3.1	IS		External test code (i.e. code used by LIS)	<OBX ID>
OBX	3.2	IS		Internal test code (i.e. code used by cobas IT 3000)	<OBX ANALYTNR>
OBX	3.3	IS		Profile code (if test was carried out as part of a profile)	<OBX PROFIL>
OBX	5.1	ST		Test result	<OBX VALUE>
OBX	5.2	ST		Validation status. This shows the validation status of the result, (normal range, ok, etc.) as shown in cobas IT 3000 in Parameter > Validation status .	<OBX VALIDSTAT>
OBX	6	ST		Units (in which result is expressed)	<OBX UNITS>
OBX	7.1	ST		Reference range: the normal or reference range set in the application for the analyte.	<OBX REFBEREICH>
OBX	7.2	ST		Lower normal range boundary	<OBX UNORMAL>
OBX	7.3	ST		Upper normal range boundary	<OBX ONORMAL>

Receiving data from cobas IT 3000

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
OBX	8	IS		Abnormal flags. Indicates that test was out of normal range. At a minimum, the following are supported: <ul style="list-style-type: none"> L = below low normal H = above high normal LL = below panic normal HH = above panic high 	<OBX INDIKATOR>
OBX	11	IS		Result status. In this case, set to F=final results.	
OBX	14	TS		Date/time test result was received by cobas IT 3000 , considered to be the time the test was carried out.	<OBX COMPLETEDATE>
OBX	15.1	IS		Instrument number	<OBX GERNR>
OBX	15.2	ST		Instrument name	<OBX GERNAME>
OBX	16.1	ST		User who released the result	<OBX RELEASEUID>
OBX	16.2	TS		Date and time the result was released	<OBX RELEASEDATE>
OBX	16.3	ST		Responsible observer. User who performed medical validation	<OBX MEDVALUID>
OBX	16.4	TS		Date the result was medically validated	<OBX MEDVALDATE>
OBX	19	TS		The instrument timestamp. This is considered to be the time of analysis.	<OBX INSTRUMENT_MEASUREMENT_TIME>
NTE	1	NM		Set ID. Sequence number of segment (numbers multiple segments of the same type)	<NTE SET>
NTE	2	IS		Indicates the source of the comment. L=normal comment; I=Instrument flag	<NTE SOURCE>
NTE	3	IS		If instrument flag: contains flag code and description sent by an instrument, in the format: Code: [code] Comment: [description] If normal comment: field contains a text comment.	<NTE CODE>, <NTE DESC>, <NTE TEXT>

(a) Data type abbreviations: ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for HL7-defined tables

Values not mapped by default

There are further values that **cobas IT 3000** can send to the host in an observation result message, but are not transferred by the default XSL templates. If you wish to use these values, you will need to edit the XSL templates to support them.

👁 For details of how the XSL templates control the data passed between **cobas IT 3000** and the host, see *Reading the ASTM or HL7 messages* on page A-10.

HL7 Segment	Pos. in HL7	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
-	-	ST		Optional	It is possible to use an alphanumeric alternative for the order number of sample number. In this case, the host should send the alphanumeric code identifying the order. Up to 30 characters are supported. This is suggested as a value for field ORC-2, component 1 "Placer Order Number", but in some circumstances, ORC-3 (Filler Order Number), or ORC-4 (Placer Group Number) might be an alternative.	<ORC HOST_ORDER_ID>
-	-	ST		Optional	The host's code identifying the physician dealing with the patient. This is suggested as a value for field PV1-7, component 1 "Attending physician code ID", but in some circumstances, PV1-8 (Referring doctor), or PV1-9 (Consulting doctor) might be an alternative.	<ORC PHYSICIAN_CODE>
-	-	ST		Optional	The name of the physician. This is suggested as a value for field PV1-7, component 2 "Attending physician name". Note that currently, cobas IT 3000 does not support dividing the name into last name, first name, title, etc.	<ORC PHYSICIAN_NAME>
-	-	ST		Optional	The dilution factor for result.	<OBX VERDFAKT>
-	-	NM		Optional	A numeric boolean flag to say if there is a comment for the result: <ul style="list-style-type: none"> 0 = no comment 1 = result comment 	<OBX COMMENT>
-	-			Optional	A graphical display of result compared to the normal range defined for the analyte.	<OBX GRAPH>
-	-	ST		Optional	The Location that handles the message, as defined in Administration > Location > [Location Name] or Administration > Client > [Client Name] .	<OBX LOCATION>
-	-	ST		Optional	This is a number that distinguishes an order from other orders sent on the same day. For each order sent during the course of a day, this number increments by 1. Every day at midnight the number is reset to zero. The number cannot be set manually, but increments up to a maximum of 999999 per day.	<OBX DAYNUMBER>
-	-	ST		Optional	The abbreviated analyte synonym, as shown in the field in Parameter > analytes/ref. ranges > Abbr.	<OBX SYNONYM>
-	-	ST		Optional	The name of the analyte.	<OBX NAME>
-	-	NM	10	Optional	The code for the analyte used by the host system. This is the value in the field of Administration > KIS > [LISDB interface] > analyte assignment: External Analyte (Out) .	<OBX HOSTCODE>

(a) Data type abbreviations: ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for HL7-defined tables

Sample message The following sample message shows a typical observation result message:

```
MSH|^~\&|PACMAN|PACMAN-LAB|PSM|PSM-LAB|20110303085741||ORU^R01|8385|P|2.4|||ER|NE||8859/1|
PID|1||0001214173||Nesbitt^Mary||19570404|W|
PV1|1|||||||||||||||||||||||||||||||||||||
ORC|1|7001002|||||||20110303085403|
OBR|1|1002||102^102|||20110303085713|||||||||||||||||||||||||||||||||||||N|
OBX|1||102^102^||5.0^ok!|mmol/L|3.7 - 5.4^3.7^5.4||||F|||20110303085713|^|BOEHLIM^03-MAR-
11^BOEHLIM^03-MAR-11|||||
OBR|2|1002||104^104|||20110303085716|||||||||||||||||||||||||||||||||||||N|
OBX|2||104^104^||2.500^ok!|mmol/L|2.200 - 2.550^2.2^2.55||||F|||20110303085716|^|BOEHLIM^03-MAR-
11^BOEHLIM^03-MAR-11|||||
```

Figure B-14 Observation result message

Receiving an acknowledgement message from an order request message

When **cobas IT 3000** is configured to send acknowledgements, it sends the host an acknowledgement message in reply to an order request message.

Message structure

Segment	Segment name	Description
MSH	Message header	General information about message, including the message type
MSA		

Configuring the interface with parameters

You can configure the host interface by using the parameters in **Workplaces > Administration > Clients > [your client] > Configuration**. The default stylesheet provides the following for messages with acknowledgement messages sent from **cobas IT 3000** in response to an order request message:

Parameter	Type	Default value	Other possible values	Instruction to cobas IT 3000:
SENDING_APPLICATION	String		(any)	The value to write in MSH-3, for the “sending application”. MSH-3. If set, this value overrides any value configured in cobas IT 3000 .
SENDING_FACILITY	String		(any)	The value to write in MSH-4, for the “sending facility”. If set, this value overrides any value configured in cobas IT 3000 .
RECEIVING_APPLICATION	String		(any)	The value to write in MSH-5, for the “receiving application”. If set, this value overrides any value configured in cobas IT 3000 .
RECEIVING_FACILITY	String		(any)	The value to write in MSH-6, for the “receiving facility”. If set, this value overrides any value configured in cobas IT 3000 .

Table B-8 Parameters used for incoming messages.

Field mappings

This list contains the HL7 fields generated by **cobas IT 3000** for an acknowledgement message to an order request, and provides a short description for each field.

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
MSH	1	ST		The delimiter, set to the bar symbol().	
MSH	2	ST		The other encoding characters. Set to ^~\&.	
MSH	3	ST		Sending application. It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH_SENDING_APPLICATION>
MSH	4	ST		Sending facility (laboratory). It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH_SENDING_FACILITY>
MSH	5	ST		Receiving application. It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH_RECEIVING_APPLICATION>
MSH	6	ST		Receiving facility (LIS). It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH_RECEIVING_FACILITY>
MSH	7	TS		Date/time message is generated	<MSH_DATE>
MSH	9.1	ID		Message Type, in this case ACK	<MSH_MESSAGE_TYPE>
MSH	9.2	ID		Message Event, in this case not used.	<MSH_EVENT_TYPE>
MSH	10	ID		Message control ID. The cobas IT 3000 echoes the value the host sent to in this field. This sends an acknowledgement to the HIS. Required if MSH.15 below is AL, SU, ER.	<MSH_CONTROL_ID>
MSH	11	ID		Processing ID. This tells the systems to use the HL7 processing rules.	<MSH_PROCESSING_ID>
MSH	12	ID		Version ID. This shows the version of HL7 used in the message.	<MSH_VERSION>
MSH	15	ID	2	Defines conditions for sending accept acknowledgment messages. Requiring accept messages only makes sense with socket-based messaging, but not with file-based messaging. <ul style="list-style-type: none"> AL = Always (Default if blank) SU = Success ER = Error NE = Never 	<MSH_ACCEPT_TYPE>
MSH	16	ID	2	Defines conditions for sending application acknowledgment messages <ul style="list-style-type: none"> AL = Always (Default if blank) SU = Success ER = Error NE = Never 	<MSH_ACK_TYPE>

Receiving data from cobas IT 3000

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
MSH	18	ID		Defines the character set used in the message. (utf-8, ISO-8859-1, etc.)	<MSH ENCODING>
MSA	1			This field contains an acknowledgment code. <ul style="list-style-type: none"> AA Original mode: Application Accept - Enhanced mode: Application acknowledgment: Accept AE Original mode: Application Error - Enhanced mode: Application acknowledgment: Error AR Original mode: Application Reject - Enhanced mode: Application acknowledgment: Reject 	<MSA ACK>
MSA	2			This field contains the message control ID of the message sent by the sending system. It allows the sending system to associate this response with the message for which it is intended.	<MSA SEQ>
MSA	3			This field contains any diagnostic text associated with the error. This could include messages from the operating system or other applications. This text is also written to the log file, where there may be more useful information.	<MSA TEXT>

(a) Data type abbreviations: ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for HL7-defined tables

Currently, other acknowledgement messages are not supported.

Receiving quality control results

Quality Control (QC) test result messages are used to send the results of quality control tests to a host (LIS).

Message structure

An **Quality Control result** contains the following segments:

Segment	Segment name	Description
MSH	Message header	General information about message, including the message type
PID	Patient identification	Nominal patient information: here identifying the test.
{		
ORC	Common order data	General information about order such as the order creator, the creation date and the order status
OBR	Observation request	Details about required test
OBX	Observation result	Test result. This includes the actual result, the unit, the reference ranges, etc.
[[NTE]]	Comments on result	Comments about the result and its interpretation
}		

Table B-9 Structure of an observation result

Configuring the interface with parameters

You can configure the host interface by using the parameters in **Workplaces > Administration > Clients > [your client] > Configuration**. The default stylesheet provides the following for messages with quality control result messages from **cobas IT 3000**:

Parameter	Type	Default value	Other possible values	Instruction to cobas IT 3000:
SENDING_APPLICATION	String		(any)	The value to write in MSH-3, for the “sending application”. MSH-3. If set, this value overrides any value configured in cobas IT 3000 .
SENDING_FACILITY	String		(any)	The value to write in MSH-4, for the “sending facility”. If set, this value overrides any value configured in cobas IT 3000 .
RECEIVING_APPLICATION	String		(any)	The value to write in MSH-5, for the “receiving application”. If set, this value overrides any value configured in cobas IT 3000 .

Table B-10 Parameters used for incoming messages.

Receiving data from cobas IT 3000

Parameter	Type	Default value	Other possible values	Instruction to cobas IT 3000:
RECEIVING_FACILITY	String		(any)	The value to write in MSH-6, for the “receiving facility”. If set, this value overrides any value configured in cobas IT 3000 .
ACCEPT_TYPE	String	AL		<p>The value to write in MSH-15, as the accept acknowledgement type:</p> <ul style="list-style-type: none"> • <i>AL</i> = Always (Default if blank) • <i>SU</i> = Success • <i>ER</i> = Error • <i>NE</i> = Never <p>Using this only makes sense with socket-based messaging. If set, this value overrides any value configured in cobas IT 3000.</p>
ACK_TYPE	String	AL		<p>The value to write in MSH-16, as the application acknowledgement type:</p> <ul style="list-style-type: none"> • <i>AL</i> = Always (Default if blank) • <i>SU</i> = Success • <i>ER</i> = Error • <i>NE</i> = Never <p>If set, this value overrides any value configured in cobas IT 3000.</p>

Table B-10 Parameters used for incoming messages.**Field mappings**

This list contains the HL7 fields generated by **cobas IT 3000** and provides a short description for each field.

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
MSH	1	ST		The delimiter, set to the bar symbol().	
MSH	2	ST		The other encoding characters. Set to ^~\&.	
MSH	3	ST		Sending application. It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH SENDING_APPLICATION>
MSH	4	ST		Sending facility (laboratory). It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH SENDING_FACILITY>
MSH	5	ST		Receiving application. It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH RECEIVING_APPLICATION>
MSH	6	ST		Receiving facility (LIS). It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH RECEIVING_FACILITY>

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
MSH	7	TS		Date/time message is generated	<MSH DATE>
MSH	9.1	ID		Message Type, in this case "ORU"	<MSH MESSAGE_TYPE>
MSH	9.2	ID		Message Event, in this case "R01"	<MSH EVENT_TYPE>
MSH	10	ID		Message control ID. The cobas IT 3000 echoes the value the host sent to in this field. This sends an acknowledgement to the HIS. Required if MSH.15 below is <i>AL</i> , <i>SU</i> , <i>ER</i> .	<MSH CONTROL_ID>
MSH	11	ID		Processing ID. This tells the systems to use the HL7 processing rules. In this case, this is set to Q = Quality control result.	<MSH PROCESSING_ID>
MSH	12	ID		Version ID. This shows the version of HL7 used in the message.	<MSH VERSION>
MSH	15	ID	2	Defines conditions for sending accept acknowledgment messages. Requiring accept messages only makes sense with socket-based messaging, but not with file-based messaging. <ul style="list-style-type: none"> <i>AL</i> = Always (Default if blank) <i>SU</i> = Success <i>ER</i> = Error <i>NE</i> = Never If using the default templates, it is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH ACCEPT_TYPE>
MSH	16	ID	2	Defines conditions for sending application acknowledgment messages <ul style="list-style-type: none"> <i>AL</i> = Always (Default if blank) <i>SU</i> = Success <i>ER</i> = Error <i>NE</i> = Never If using the default templates, it is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH ACK_TYPE>
MSH	18	ID		Defines the character set used in the message. (utf-8, ISO-8859-1, etc.)	<MSH ENCODING>
PID	1	NM		Set ID. Sequence number of segment (numbers multiple segments of the same type).	<PID SET>
ORC	1	NM		Set ID. Sequence number of segment (numbers multiple segments of the same type)	<ORC SET>
ORC	2	NM		The Host's ID for the analyte or test.	<ORC CONTROL_HOST_ID>
ORC	5	ID		Status of the order. Set to <i>IP</i> , standing for "in progress".	<>

Receiving data from cobas IT 3000

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
ORC	7.6	ID		Priority. Set to R, standing for "routine".	<>
OBR	1	NM		Set ID. Sequence number of segment (numbers multiple segments of the same type)	<OBR SET>
OBR	15.1	ST		Specimen	<ORC SPECIMEN>
OBR	15.7	ID		Specimen Role. Set to Q, standing for "Quality Control".	
OBR	20	ST		Instrument identifier	<ORC INSTRUMENT>
OBR	27.6	ST		Priority. Set to R, standing for "Routine".	
OBX	1	NM		Set ID. Sequence number of segment (numbers multiple segments of the same type)	<OBX SET>
OBX	3.1	IS		External test code (i.e. code used by LIS)	<OBX ID>
OBX	3.2	IS		Internal test code (i.e. code used by cobas IT 3000)	<OBX ANALYTNR>
OBX	3.3	IS		Profile code (if test was carried out as part of a profile)	<OBX PROFIL>
OBX	5.1	ST		Test result	<OBX VALUE>
OBX	5.2	ST		Validation status. Indicates if result has been validated	<OBX VALIDSTAT>
OBX	6	ST		Units (in which result is expressed)	<OBX UNITS>
OBX	7.1	ST		Reference range: the normal or reference range set in the application for the analyte.	<OBX REFBEREICH>
OBX	7.2	ST		Lower normal range boundary	<OBX UNORMAL>
OBX	7.3	ST		Upper normal range boundary	<OBX ONORMAL>
OBX	8	IS		Abnormal flags. Indicates that test was out of normal range. At a minimum, the following are supported: <ul style="list-style-type: none"> L = below low normal H = above high normal LL = below panic normal HH = above panic high 	<OBX INDIKATOR>
OBX	11	IS		Result status. In this case, set to F=final results.	
OBX	14	TS		Date/time test was carried out	<OBX COMPLETEDATE>

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
OBX	15.2	ST		Instrument name	<OBR GERNAME>
OBX	15.4	ST		Instrument name	<OBR INSTRUMENT>
OBX	18	ST		Instrument name	<OBR INSTRUMENT>
OBX	19	TS		The instrument timestamp. This is considered to be the time of analysis.	<OBX INSTRUMENT_MEASUREMENT_TIME>
NTE	1	NM		Set ID. Sequence number of segment (numbers multiple segments of the same type)	<NTE SET>
NTE	2	IS		Indicates the source of the comment. <ul style="list-style-type: none"> └ = Normal comment; ┐ = Instrument flag 	<NTE SOURCE>
NTE	3	IS		If instrument flag: contains flag code and description sent by an instrument, in the format: Code: [code] Comment: [description] If normal comment: field contains a text comment.	<NTE CODE>, <NTE DESC>, <NTE TEXT>

(a) Data type abbreviations: ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for HL7-defined tables

Values not mapped by default

There are further values that **cobas IT 3000** can send to the host with a quality control result, but are not transferred by the default XSL templates. If you wish to use these values, you will need to edit the XSL templates to support them.

👁 For details of how the XSL templates control the data passed between **cobas IT 3000** and the host, see *Reading the ASTM or HL7 messages* on page A-10.

HL7 Segment	Pos. in HL7	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
-	-	ST		Optional	This is the name of the control, for example PNU or PPU. This is a suggested value for the Specimen Segment (SPM), field 2, component 1, (instead of the patient sample number).	<ORC CONTROL_NAME>
-	-	NM		Optional	This is the lot number of the control material. This is a suggested value for the Specimen Segment (SPM), field 2, component 2, (instead of the patient sample number).	<ORC CONTROL_LOT>

Receiving data from cobas IT 3000

HL7 Segment	Pos. in HL7	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
-	-	IS		Optional	This is the ID number used by the host to identify the control. This is a suggested value for the Specimen Segment (SPM), field 2, component 3, (instead of the patient sample number).	<ORC CONTROL_HOST>
-	-	IS		Optional	An alternative ID number used by the host to identify the control. This is a suggested value for the Specimen Segment (SPM), field 2, component 3, (instead of the patient sample number).	<ORC CONTROLID>
-	-	ST		Optional	This is ID for the instrument and the module that performed the quality control test. This is suggested as a value for the Observation Result Segment (OBX), field 18.	<OBX GERNAME>

(a) Data type abbreviations: ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for HL7-defined tables

Sample Event message

A Sample Event message tells a host (LIS) that a particular sample has been located, and tells the host about the sample's current status.

Message structure

A **Sample Event** message contains the following segments:

Segment	Segment name	Description
MSH	Message header	General information about message, including the message type
EQU	Equipment Detail Segment	Data necessary to identify and maintain the equipment that is being used throughout the Laboratory Information System, and to notify the host or Hospital Information System of the status of the sample.
SAC	Container Detail Segment	Data necessary to maintain the containers, such as sample tubes, that are being used throughout the Laboratory Information System.

Table B-11 Structure of a Sample Event message

Configuring the interface with parameters

You can configure the host interface by using the parameters in **Workplaces > Administration > Clients > [your client] > Configuration**. The default stylesheet provides the following for messages with sample event messages from **cobas IT 3000**:

Parameter	Type	Default value	Other possible values	Instruction to cobas IT 3000:
SENDING_APPLICATION	String		(any)	The value to write in MSH-3, for the "sending application". MSH-3. If set, this value overrides any value configured in cobas IT 3000 .
SENDING_FACILITY	String		(any)	The value to write in MSH-4, for the "sending facility". If set, this value overrides any value configured in cobas IT 3000 .
RECEIVING_APPLICATION	String		(any)	The value to write in MSH-5, for the "receiving application". If set, this value overrides any value configured in cobas IT 3000 .
RECEIVING_FACILITY	String		(any)	The value to write in MSH-6, for the "receiving facility". If set, this value overrides any value configured in cobas IT 3000 .
SEND_ONCE	String			Not used in default template
SEND_INSTRUMENTS	String	ALL		Not used in default template

Table B-12 Parameters used for incoming messages.

Receiving data from cobas IT 3000

Parameter	Type	Default value	Other possible values	Instruction to cobas IT 3000:
SEND_LOCATION	String	ALL		Not used in default template
ACCEPT_TYPE	String	AL		<p>The value to write in MSH-15, as the accept acknowledgement type:</p> <ul style="list-style-type: none"> • <i>AL</i> = Always (Default if blank) • <i>SU</i> = Success • <i>ER</i> = Error • <i>NE</i> = Never <p>Using this only makes sense with socket-based messaging. If set, this value overrides any value configured in cobas IT 3000.</p>
ACK_TYPE	String	AL		<p>The value to write in MSH-16, as the application acknowledgement type:</p> <ul style="list-style-type: none"> • <i>AL</i> = Always (Default if blank) • <i>SU</i> = Success • <i>ER</i> = Error • <i>NE</i> = Never <p>If set, this value overrides any value configured in cobas IT 3000.</p>

Table B-12 Parameters used for incoming messages.**Field mappings**

This list contains the HL7 fields generated by **cobas IT 3000** and provides a short description for each field.

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
MSH	1	ST		The delimiter, set to the bar symbol().	
MSH	2	ST		The other encoding characters. Set to ^~\&.	
MSH	3	ST		Sending application. It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH SENDING_APPLICATION>
MSH	4	ST		Sending facility (laboratory). It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH SENDING_FACILITY>
MSH	5	ST		Receiving application. It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH RECEIVING_APPLICATION>
MSH	6	ST		Receiving facility (LIS). It is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH RECEIVING_FACILITY>

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
MSH	7	TS		Date/time message is generated	<MSH DATE>
MSH	9.1	ID		Message Type, in this case SSU.	<MSH MESSAGE_TYPE>
MSH	9.2	ID		Message Event, in this case U03.	<MSH EVENT_TYPE>
MSH	10	ID		Message control ID. The cobas IT 3000 uses this to send an acknowledgement to the HIS.	<MSH CONTROL_ID>
MSH	11	ID		Processing ID. Set to P = "Production".	<MSH PROCESSING_ID>
MSH	12	ID		Reserved field for future development, to show version of cobas IT 3000 .	<MSH VERSION>
MSH	13	NM	15	Sequence number (if Sequence number protocol is to be used)	<MSH SEQ>
MSH	15	ID	2	Accept Acknowledgment Type. Defines conditions for sending accept acknowledgment messages. Requiring accept messages only makes sense with socket-based messaging, but not with file-based messaging. Permitted values are: <ul style="list-style-type: none"> AL = Always NE = Never ER = Error/reject conditions only SU = Successful completion only If using the default templates, it is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH ACCEPT_TYPE>
MSH	16	ID	2	Application Acknowledgment Type. Defines conditions for sending application acknowledgment messages Permitted values are: <ul style="list-style-type: none"> AL = Always NE = Never ER = Error/reject conditions only SU = Successful completion only If using the default templates, it is possible for the user to set this in cobas IT 3000 Administration > Client > [Client name] > [Connection name] > Configuration.	<MSH ACK_TYPE>
MSH	18	ID		Defines the character set used in the message.	<MSH ENCODING>
EQU	1.4	ST		The name of the instrument. In the case of an archive message, this field shows instead the user logged into cobas IT 3000 . (Optional).	<EQU EQUIPMENT_IDENTIFIER_NAME>
EQU	1.5	NM		Location, as defined in cobas IT 3000 , from which the sample came, such as the hospital or laboratory. (Optional).	<EQU EQUIPMENT_IDENTIFIER_CLIENT>
EQU	1.6	ST		The function call. Set to SAMPLEEVENT.	<EQU EQUIPMENT_IDENTIFIER_MODULE>

Receiving data from cobas IT 3000

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
EQU	1.7	IS		<p>The content of the message, identifying the event. This parameter is optional. Permitted values are:</p> <ul style="list-style-type: none"> SEEN This identifies a Sample viewed message. DISPOSE This identifies a Sample disposed of message. LOESCH This identifies a Sample deleted message. NEW This identifies a Sample entered message. RESULT This identifies a Sample analyzed message. RETRIEVE This identifies a Sample retrieved message. SCAN This identifies a Sample scanned message. ARCHIVE This identifies a Sample archived message. 	
					<EQU EQUIPMENT_IDENTIFIER_EVENT>
EQU	2	TS		This field is the date and time that the event (e.g., state transition, issuing of command, finishing of command execution) occurred. (Format: YYYYMMDDHHMMSS).	
					<EQU EVENT_DATE_TIME>
SAC	3	ST		Sample id. Identifier for the sample or specimen. For an aliquot, this contains the identifier for the aliquoted sample or specimen. (Optional).	
					<SAC CONTAINER_IDENTIFIER>
SAC	4	ST		Parent sample id. For an aliquot, the identifier for the parent sample or specimen from which the aliquot was made. In other cases, this field is empty.	
					<SAC PRIMARY_CONTAINER_IDENTIFIER>
SAC	6	ST		Specimen Type. Descriptor of the specimen, for example SE or 01 for Serum.	
					<SAC SPECIMEN_SOURCE>
SAC	7	TS		This field is the date and time that the container was last registered with the Laboratory Information System, e.g., reading of a container bar code by a device. (Format: YYYYMMDDHHMMSS).	
					<MSH DATE>

HL7 Segment	Position	Data type ^(a)	Max length	Description	XML tag/Attribute
SAC	8.1	ID		<p>This field contains a single character HL7 code that identifies the status of the unique container in which the specimen resides at the time that the transaction was initiated.</p> <ul style="list-style-type: none"> I Identified. The container has been received. P In Position. The container is in position for specimen transfer (e.g., container removal from track, pipetting, etc.). O In Process. The container is being processed by the equipment. It is useful as a response to a query about Container Status, when the specific step of the process is not relevant. R Process Completed. Processing has been completed, but the container has not been released. L Left Equipment. The container has been released from that system. M Missing. The container did not arrive at its next expected location. X Container Unavailable. The container is no longer available within the scope of the system (e.g., tube broken or discarded). U Unknown. The container has not been identified. 	<SAC CONTAINER_STATUS>
SAC	8.4			The error code returned by the instrument, if applicable.	<SAC SPECIFIC_CONTAINER_STATUS>
SAC	9			The carrier type, for example the type of the tray, e.g. ARCHIVE, or the size of the tray e.g. 20x5.	<SAC CARRIER_TYPE>
SAC	10			The tray number, of the sample, as the first part of the Instrument Specimen ID.	<SAC CARRIER_IDENTIFIER>
SAC	11		5	The position of the sample on the tray, as the second part of the Instrument Specimen ID. If the rack is configured in columns and rows, this field shows only the column. Otherwise, it shows the number that identifies the position of the sample.	<SAC POSITION_IN_CARRIER>

(a) Data type abbreviations: ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for HL7-defined tables

Receiving data from cobas IT 3000

Example Sample Event message This message notifies the host that the **cobas IT 3000** has seen a sample in an archive rack:

```
MSH|^&\|CIT3000|LAB|PSM|LAB|||20020606161400||SSU^U03|20020606161402|P|2.4|||ER|ER||8859/1|
EQU|^RSD800A^80^SAMPLEEVENT^SEEN|20000215094700
SAC|||100000025001|100000025001||01|20000215094700|P|ARCHIVE|ARCHIVE- 1|1|
```

Figure B-15 Sample Seen message.

This message notifies the host that the **cobas IT 3000** has seen a sample on a cobas 6000 rack:

```
MSH|^&~\|ROCHE|ROCHE-LAB|PSM|PSM-LAB|20090605165726||SSU^U03|2116273|P|2.4|||NE|NE||8859/1|
EQU|^ODIN^LAB1^SAMPLEEVENT^SEEN|20090605165726|
SAC|||100000103|||01|20090605165726|P^^^|cobas 6000|0230|1^|
```

Figure B-16 Sample Seen message.

This message notifies the host that a sample has been deleted on the **cobas IT 3000**.

```
MSH|^&~\|ROCHE|ROCHE-LAB|PSM|PSM-LAB|20100818151448||SSU^U03|16238|P|2.4|||NE|NE||8859/1|
EQU|^BOEHLIM^LAB1^SAMPLEEVENT^DELETE|20100818151448|
SAC|||50030099|||01|20100818151448|X^^^|||
```

Figure B-17 Sample Delete message

This message notifies the host that a sample has been scanned on the **cobas IT 3000**.

```
MSH|^&~\|ROCHE|ROCHE-LAB|PSM|PSM-LAB|20100818151654||SSU^U03|16259|P|2.4|||NE|NE||8859/1|
EQU|^BOEHLIM^LAB1^SAMPLEEVENT^SCAN|20100818151654|
SAC|||303101|||01|20100818151654|P^^^|||
```

Figure B-18 Sample Scan message

This message notifies the host that a new sample has been entered on the **cobas IT 3000**.

```
MSH|^&~\|ROCHE|ROCHE-LAB|PSM|PSM-LAB|20100818151824||SSU^U03|16278|P|2.4|||NE|NE||8859/1|
EQU|^BOEHLIM^LAB1^SAMPLEEVENT^NEW|20100818151824|
SAC|||88899901|||01|20100818151824|I^^^|||
```

Figure B-19 Sample new message

This message notifies the host that the **cobas IT 3000** has archived a sample.

```
MSH|^&~\|ROCHE|ROCHE-LAB|PSM|PSM-LAB|20100818152318||SSU^U03|16298|P|2.4|||NE|NE||8859/1|
EQU|^BOEHLIM^LAB1^SAMPLEEVENT^ARCHIVE|20100818152318|
SAC|||88899901|||01|20100818152318|P^^^|Man_ARCH_SER|Man_ARCH_SER-49|2^|
```

Figure B-20 Sample Archive message

ASTM interface

C

7	<i>ASTM protocol (LIS2 - A2)</i>	C-3
8	<i>Description of the ASTM Interface (LISDB)</i>	C-13

ASTM protocol (LIS2 - A2)

The low-level definition of ASTM

This chapter presents the lower layers of the ASTM protocol, as used by **cobas IT 3000**.

In this chapter	Chapter 7
Background to the ASTM protocol	C-5
Communication processing layers	C-6
ASTM lower layer	C-7
ASTM syntax	C-8
Coding rules for the messages	C-8
End of Record character	C-8
Field Delimiter = vertical bar ' '	C-8
Repeat Delimiter = backslash '\'	C-8
Component Delimiter = caret '^'	C-8
Escape Character = ampersand '&'	C-9
Expression of special characters with Escape Character	C-9
Message Transmission Phases	C-9
Checksum Calculation / Message Frame	C-11

Background to the ASTM protocol

ASTM (American Society of Testing and Materials) has a plan for communications between automatic analyzers and host computers for standards E1381-91 (Specification for Low-Level Protocol to Transfer Messages Between Clinical Laboratory Instruments and Computer Systems) and E1394-91 (Standard Specifications for Transferring Information Between Clinical Instruments and Computer Systems). The basic specifications of the standards are regulated on X12 of ANSI.

Communication processing layers

The communication process between the system and the host is divided into four layers as shown below. This specification explains the processing and operation methods for the application layer.

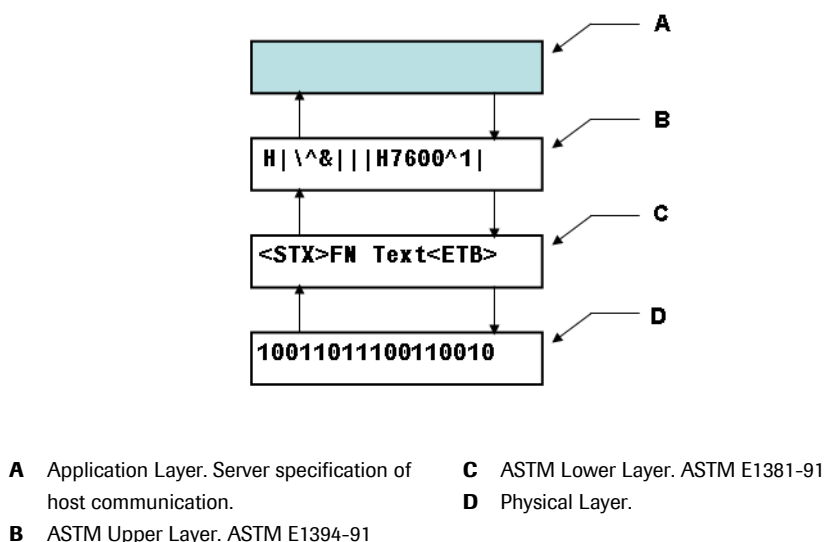


Figure C-1 Host Communication Processing Layers

Details of the ASTM protocol can be found in the *Annual Book of ASTM Standards*. Copyright American Society for Testing and Materials, 100 Barr Harbor Drive, West Conshohocken, PA 19428-2959, USA.

- ASTM E1381-91 Low Level Protocol
Specification for Low Level Protocol to Transfer Messages Between Clinical Laboratory Instruments and Computer Systems
- ASTM E1394-91 High Level Protocol
Standard Specification for Transferring Information Between Clinical Instruments and Computer Systems

ASTM lower layer

ASTM lower layer receives messages for a transmission request from the upper layer. These messages are then split into frames and sent to a communication medium to be transmitted to other parties. ASTM lower layer also constructs frames received from a communication medium to recreate messages to be transferred to the ASTM upper layer as reception messages. Configuration and communication procedures for transmission and reception of frames are explained in the following sections.

Item	Method	Explanation
Frame Configurations	For Middle Frame	<ul style="list-style-type: none"> Control character (characters enclosed in <>):
	<STX> FN text <ETB> C1 C2 <CR><LF>	<STX> is control character (HEX 02) <ETB> is control character (HEX 17)
	For Last Frame	<CR> is control character (HEX 0D)
	<STX> FN text <ETX> C1 C2 <CR><LF>	<LF> is control character (HEX 0A) <ETX> is control character (HEX 03)
		<ul style="list-style-type: none"> FN: Frame Number. The frame number is an ASCII digit ranging from 0 to 7. The frame number begins at 1 with the first frame of the Transfer phase. The frame number is incremented by one for every new frame transmitted. After 7, the frame number rolls over to 0, and continues in this fashion. (i.e. modulus 8) Text: the data content of a frame (maximum 240 characters). Records are subdivided into intermediate (middle) frames with 240 or fewer characters. Text is part of a split message. C1 and C2: Checksum as a two-digit hexadecimal number. The checksum permits the receiver to detect a defective frame. The checksum is encoded as two characters which are sent after the <ETB> or <ETX> character. The checksum is computed by adding the binary values of the characters, keeping the least significant eight bits of the result. C1 = most significant character of checksum 0 to 9 and A to F C2 = least significant character of checksum 0 to 9 and A to F
Frame Character Configuration of Text	Characters other than	<SOH> is control character (HEX 01)
	<SOH><STX><ETX>	<EOT> is control character (HEX 04)
	<EOT><ENQ><ACK>	<ENQ> is control character (HEX 05)
	<DLE><NAK><SYN>	<ACK> is control character (HEX 06)
	<ETB><CR><LF>	<DLE> is control character (HEX 10)
	<DC1><DC2><DC3><DC4>	<NAK> is control character (HEX 15) <SYN> is control character (HEX 16)
Maximum Length of the Frame	247 characters	<DC1> ~ <DC4> are control characters (HEX 11 ~ 14)
		For one frame, maximum of 240 characters for text, plus 7 characters for frame control characters. Messages equal to or less than 240 characters are transmitted as one final frame. Messages greater than 240 characters are split into frames that have character lengths that fall within the 240-character limit. The only or final remaining frame becomes the last frame and is indicated by <ETX>. All others are intermediate (middle) frames and are indicated by <ETB>.
		Intermediate Frame
		<STX> FN text <ETB> C1 C2 <CR><LF>
		Last Frame
		<STX> FN text <ETX> C1 C2 <CR><LF>

ASTM syntax

The structure of the messages to be transferred, according to ASTM Communication Regulation, is explained in this section. Between **cobas IT 3000** and the host, various data such as Test Requests and Results are transferred back and forth. All of these data conform to this syntax.

Message	A message is constructed with an arrangement of several records (refer to the next item). It is the smallest unit of information transferred between a host and an analyzer. Messages begin with a 'Message Header Record' that indicates the beginning of a message and end with a 'Message Termination Record' that indicates the end of a message.
Record	A record is constructed from several fields and expresses a single purpose (such as to specify result reports or test requests). A record may be repeated or used singularly in a message. Code that indicates the purpose of a record is noted in the first character of that record.
Field	A field is the ASTM's smallest element to construct information. Attributes for a field (name, format, and meanings) are defined in units in a record.

Coding rules for the messages

This section deals with message coding rules as well as special characters, such as delimiters, used to develop messages provided by records and fields.

End of Record character

The ASCII CR character (HEX 0D) is always used to indicate the end of a record.

Field Delimiter = vertical bar '|'

A Field delimiter is a character used to separate fields that are next to each other in a record. This is also a delimiter for the first Record ID (character that appears in the beginning of a record) and the next field. According to the 2nd character that appears in the Message Header Record (record that appears in the front of a message), a Field delimiter can be defined with an optional character through the Message Header Record; however, it is recommended that a vertical bar '|' be used.

Repeat Delimiter = backslash '\'

When a field is constructed by the same data repeated several times, it is referred to as a Repeated Field. The delimiter between the repeated items for the Repeated Field is called the Repeat delimiter. Repeat delimiters can be defined with an optional character through the Message Header Record; however, it is recommended that a backslash '\' be used.

Component Delimiter = caret '^'

When a field is constructed by several elements, it is referred to as a Component Field. The delimiter between these elements is the Component delimiter. The

Component delimiter can be defined with an optional character through the Message Header Record; however, it is recommended that a caret '^' be used.

Escape Character = ampersand '&'

An Escape character is provided to indicate a delimiter for the fields that include general text. When this character occurs in a relevant field, the next character holds a special meaning (discussed below). An Escape character can be defined with an optional character through the Message Header Record, however, it is recommended that an ampersand '&' be used.

Expression of special characters with Escape Character

The following escape sequence (starting with & and ending with &) is defined. When this sequence is detected in a field, it is changed to a corresponding character and deleted.

Escape sequences other than these are skipped and treated as NULL values.

&F&	Indicates field delimiter
&S&	Indicates component delimiter
&R&	Indicate Repeat delimiter
&E&	Indicates Escape

Message Transmission Phases

To establish which system sends and which system receives information and to assure the actions of sender and receiver are well coordinated, there are three distinct phases in transferring information.

- Establishment phase
- Transfer phase
- Termination phase

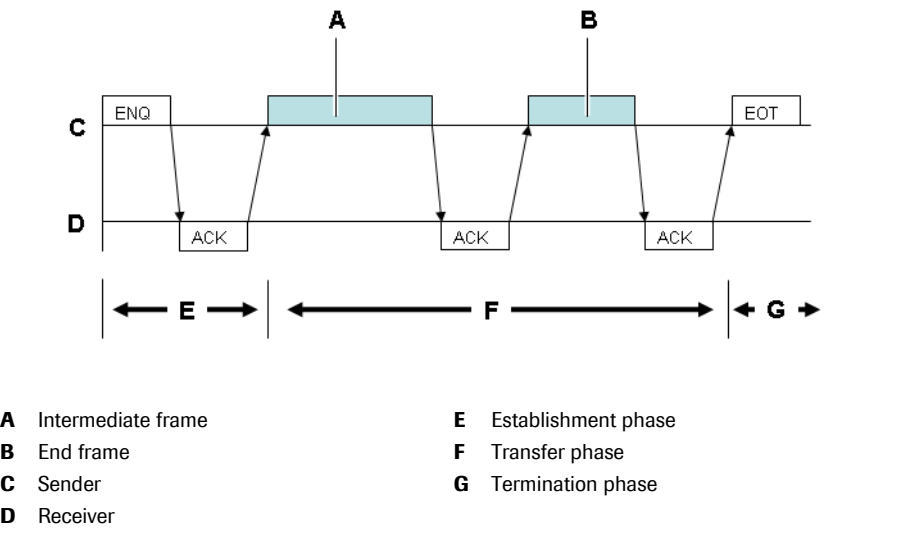


Figure C-2 Message Transaction Phases

Within the transfer phase, all records of the corresponding message are grouped into longer frames to increase speed. The records are separated through a [CR] character.

ASTM syntax

Therefore, to obtain pure ASTM records again, the receiver must concatenate all the frames and wait for a [EOT] character. Then, the receiver can process the frame and split it into different records using the [CR] as the separator.

Checksum Calculation / Message Frame

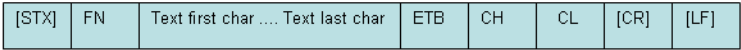


Figure C-3 The intermediate frame

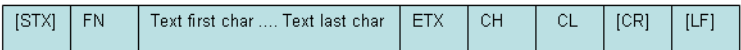


Figure C-4 The end frame

[STX]	The ASCII code 2, indicating the beginning of a frame transmission.
FN	The frame number modulus 8. Frames of a single Transmission Phase are consecutively numbered beginning with 1. So FN runs from 1 to 7, continues with 0, 1, and so on. Use ASCII codes for the digits '0' to '7' (48-55).
Text	The data content of a frame (max. 240 characters). Records are sub-divided into intermediate frames with 240 characters. Maximum is indicated by [ETB]. The only or last remaining frame is indicated by [ETX]. Different records must be sent in different frames.
[ETB]	The ASCII code 23 (17hex), indicating the end of the text block of an intermediate frame.
[ETX]	The ASCII code 3, indicating the end of the text block of an end frame.
CH, CL	Represents the high nibble (= most significant 4 bit) respectively, the low nibble (=least significant 4 bit) of the 8-bit checksum. CH and CL are represented as two digits of hex numbers. The checksum is the modulus 8 of the sum of ASCII values of the frame characters starting with and including 'FN' and completing with [ETX] respectively [ETB].

Example for Checksum
calculation

[STX]1Test[ETX]

Character	Value (hex)	Sum (hex)
[STX]	02	00
'1'	31	31
'T'	+54	85
'e'	+65	EA
's'	+73	15D
't'	+74	1D1
[ETX]	+03	1D4
	= 1D4	
	Mod 100	
	= D4	

Thus the message to be sent is:

[STX]1Test[ETX]D4[CR][LF]

Description of the ASTM Interface (LISDB)

This part of the document shows the different message types available to communicate between a Laboratory Information System (LIS) and **cobas IT 3000** solution as well as between instruments and **cobas IT 3000** solution using the ASTM protocol.

In this chapter

Chapter 8

ASTM communications	C-15
Message flow	C-15
Batch mode	C-16
ASTM message structure	C-17
Query message	C-18
Message structure	C-18
Field mappings	C-18
H - Message Header Record (Level 0)	C-18
Q - Message Query Record (Level 2)	C-19
L - Message Terminator Record (Level 0)	C-19
Order requests (and patient data)	C-20
Message structure	C-20
Configuring the interface with parameters	C-21
Field mappings	C-21
H - Message Header Record (Level 0)	C-21
P - Patient Record (Level 1)	C-22
O - Order Record (Level 2)	C-22
L - Message Terminator Record (Level 0)	C-24
Values not mapped by default	C-24
Example messages	C-25
Handling different actions in the order	C-27
Update delete level	C-28
Automatic creation of orderer	C-29
The fields available in the ORDERER element	C-31
Test results	C-33
Message structure	C-33

Configuring the interface with parameters	C-33
Field mappings	C-33
H - Message Header Record (Level 0)	C-34
P - Message Patient Record (Level 1)	C-34
O - Message Order Record (Level 2)	C-35
R - Message Result Record (Level 2)	C-35
C - Message Comment Record (Level 2)	C-36
L - Message Terminator Record (Level 0)	C-36
Values not mapped by default	C-37
Quality control results	C-39
Message structure	C-39
Configuring the interface with parameters	C-39
Field mappings	C-39
H - Message Header Record (Level 0)	C-39
P - Message Patient Record (Level 1)	C-40
O - Message Order Record (Level 2)	C-40
R - Message Result Record (Level 2)	C-40
L - Message Terminator Record (Level 0)	C-41
Values not mapped by default	C-42
Sample Event message	C-43
Message structure	C-43
Configuring the interface with parameters	C-43
Field mappings	C-44
H - Message Header Record (Level 0)	C-44
P - Message Patient Record (Level 1)	C-45
O - Message Order Record (Level 2)	C-45
M 1 EQU - Equipment Detail Record (Level 3)	C-46
M 1 SAC - Container Detail Record (Level 3)	C-47
L - Message Terminator Record (Level 0)	C-48

ASTM communications

cobas IT 3000 solution can manage a group of instruments for a laboratory information system (LIS). The **cobas IT 3000** solution's module connects as a client to the LIS server, and communicates over TCP/IP. Thus the **cobas IT 3000** solution provides a single interface through which the LIS can communicate with a group of instruments.

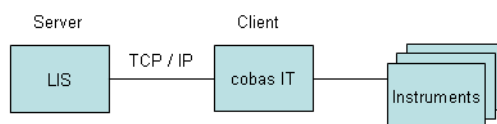


Figure C-5 **cobas IT 3000** solution managing a group of instruments.

The LIS system can send and receive messages in ASTM format to and from the **cobas IT 3000** solution. **cobas IT 3000** solution handles all further communication with its group of instruments.

Message flow

cobas IT 3000 solution handles a number of basic message types, which normally occur in the following order.

1. Query message.

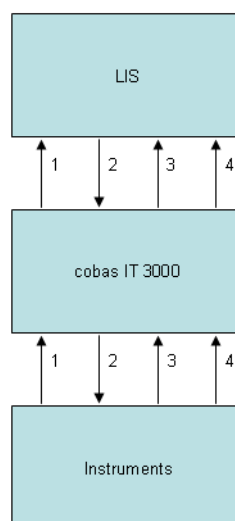
An instrument sends this message to the LIS. It says that the instrument is waiting for orders information, and what kinds of orders it is waiting for.

2. Order.

LIS sends this message to an instrument, along with a sample. It tells the instrument what test the LIS wants done on the sample.

3. Order result.

An instrument sends this message to the LIS. It gives the results of the test done on the sample.

**A** Query**B** Order**C** Result**D** Quality control or Sample Event messages**Figure C-6** The order of message flows.*Other messages*

Other messages which may occur at specific times include:

- Quality control result.

An instrument sends this message to the LIS. It notifies the LIS of the status of the quality checks done on the instrument. On the instrument or in **cobas IT 3000** solution, it is possible to configure the time when Quality Control tests are performed.

- Sample Event message.

The Sample Event message notifies the LIS that a certain sample is present on the instrument or in pre-analytic device or an archive, but without making a query for an order.

Sample Event example

cobas IT 3000 solution uses the Sample Event message to notify the LIS of the presence of the location of a sample, without making an order query. A typical example is the following scenario:

1. LIS/HIS sends sample and order information to **cobas IT 3000** solution.
2. The laboratory staff place the sample on the analyzer.
3. The analyzer queries **cobas IT 3000** solution for details of tests to be done on the sample, sending the sample ID, rack number and position.
4. **cobas IT 3000** solution replies to the analyzer with details of the tests to be done.

cobas IT 3000 solution sends a 'Sample Event' message to the LIS, notifying the LIS of the sample's rack number and position.

Batch mode

In batch mode, the LIS sends orders without waiting for an instrument to send a Query message. The **cobas IT 3000** solution stores the orders until the appropriate instrument registers, and then forwards the order to it.

ASTM message structure

This section describes the structure and use of ASTM messages.

ASTM is a protocol defined by the Clinical and Laboratory Standards Institute (formerly NCCLS) to provide a method for the two-way digital transmission of remote requests and results between clinical laboratory instruments and information systems.

The standard is described in the publication, Specification for Transferring Information Between Clinical Laboratory Instruments and Information Systems; Approved Standard—Second Edition. NCCLS document LIS2-A2 (ISBN 1-56238-550-X). NCCLS, 940 West Valley Road, Suite 1400, Wayne, Pennsylvania 19087-1898 USA, 2004.

Message structure An ASTM message contains the following segments:

Segment	Segment name	Description
H	Message header record	General information about message, including the message type
{ ^(a)		
^(b) P	Patient record	Patient information such as ID, name and first name, etc. (incl. information about the visit)
{O	Order record	General information about order and the requested tests
{[R]	Result record	Test result
[C]]]}	Comment record	Comments about the result and its interpretation
{[Q]}	Query record	Requests information or orders from a remote system.
}		
L	Terminator record	

Table C-1 Structure of an ASTM message

- (a) {} = segments which can be repeated.
- (b) [] = optional segments.

Query message

Roche Diagnostics **cobas IT 3000** sends this message to the laboratory information system (LIS). The message says that **cobas IT 3000** is ready to take order requests of a specified kind.

This message will only be sent when the option QUERY_ALWAYS=YES is set in the instrument.ini file. Then, the instrument queries **cobas IT 3000** for tests and **cobas IT 3000** forwards the request to the host. For further details, see *The cobas IT 3000 Service Manual*.

Message structure

An Query message contains the following segments:

Segment	Segment name	Description
H	Message header record	General information about message, including the message type
Q	Query record	Requests information or orders from a remote system.
L	Terminator record	Ends the message.

Table C-2 Structure of a QC result message

Field mappings

This list contains the ASTM fields generated by **cobas IT 3000** and provides a short description for each field.

H - Message Header Record (Level 0)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier for Header record (H)	
2			Delimiter definition. The following standard delimiters are recommended: \ ^ &	
	Char	1	: Field delimiter = vertical bar [124]	
	Char	1	\: Repeat delimiter = backslash [92]	
	Char	1	^: Component delimiter = caret [94]	
	Char	1	&: Escape delimiter = ampersand [38]	
5	ST		Sending application	<MSH SENDING_APPLICATION>
10	ST		Receiving application	<MSH RECEIVING_APPLICATION>
12	Char	1	Processing ID, which indicates how the message is to be processed. Only P = "Production" is currently supported.	
14	ST		Date message generated, in YYYYMMDDHHMMSS format.	<MSH DATE>

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

Q - Message Query Record (Level 2)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier that starts a Query record (Q)	
2	NM		Identifier for Query records. This is set to the digit 1 for the first record, and then increments by one for each record in the message. In this case, set to 1.	
3.2	ST		The sample ID. This is the information system specimen ID number. Note that multiple sample IDs can be sent using the repeat delimiter (backslash: \).	<QRD SAMPLEID>
5	ST		Universal Test ID. This is set to ALL . This requests all tests for the specimens specified in section 3.2.	
13	ST		The Request Information Status Code. This is set to the letter O , signifying “requesting orders and demographics”.	

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

L - Message Terminator Record (Level 0)

This is the last record, and identifies the end of the message.

Position in record	Data type ^(a)	Max length	Description
1	Char	1	Identifier for Terminator record (L)
2	Char	1	A row counter. Always set to 1, as there is only ever one Terminator record.
3	ID	1	Code that gives an explanation of the reason for ending the message. Set to N , signifying normal termination.

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

In this example, the **cobas IT 3000** says it is ready to take an order.

```
H|\^&|||P242|||GWI-LIS|P||20080110161156<CR>
Q|1|^10000072||ALL|||O<CR>
L|1|N
```

Figure C-7 Sample query message.

Order requests (and patient data)

Order requests transfer the following data from a host (LIS) to **cobas IT 3000**:

- New orders (test requests)
- Changes to an existing order
- Deletion of an existing order
- Patient data (with every order request, patient data included in the order is updated and patients who do not yet exist are created)



Danger of patient data inconsistency received from host.

Do not use this application as a main repository for patient data.

Make sure that the host (LIS/HIS) vendor always sends a unique patient ID for each patient to **cobas IT 3000**.

Any patient demographic changes coming from the host will be updated, no matter if there are results already validated and sent to the host for that specific patient. This may lead to potential wrong validation flags for the affected patient results.

Avoid this kind of change in patient demographic data.



Patient ID must be ASCII 7 string

The patient ID must be an ASCII 7 string. Whitespace, extended ASCII or unicode is not supported.

Message structure

An Order request contains the following segments:

Segment	Segment name	Description
H	Message header record	General information about message, including the message type (mandatory)
P	Patient record	Information about patient and visit (mandatory)
O	Order record	General information about order and required test(s) (mandatory)
[R]	Result record	Any results already available for the order and required test(s) (optional)
[C]	Comment record	Comments on the order and required test(s) (optional)
L	Terminator record	Indicates the end of a message

Table C-3 Structure of an order request

Configuring the interface with parameters

You can configure the host interface by using the parameters in **Workplaces > Administration > Clients > [your client] > Configuration**. The default stylesheet provides the following for test request messages sent to **cobas IT 3000**:

Parameter	Type	Default value	Other possible values	Instruction to cobas IT 3000:
acks	Boolean	true()	false()	Respond to application and accept acknowledgement instructions. (Not used in the default templates.)
patid	Integer	'3'	'4', '5'	Which field in the P record segment contains the patient ID.
sampleid	Integer	'3'	'4'	Which field in the O segment contains the sample ID.
ordernr	Integer	'3'	'4'	Which field in the O segment contains the order number. By default, cobas IT 3000 only reads the first ten digits of this field.

Table C-4 Parameters used for incoming messages.

Integer and string values must be in single quotation marks. Otherwise Oracle may behave inconsistently.

Field mappings

This list contains all supported ASTM fields. By default, all other fields contained in an ASTM order request are ignored.

H - Message Header Record (Level 0)

Position in record	Data type ^(a)	Max length	Required/optimal?	Description	XML tag/Attribute
1	Char	1	Required	Identifier for Header record (H)	
2				Delimiter definition. The following standard delimiters are recommended: \ ^ &	
	Char	1	Required	: Field delimiter = vertical bar [124]	
	Char	1	Required	\: Repeat delimiter = backslash [92]	
	Char	1	Required	^: Component delimiter = caret [94]	
	Char	1	Required	&: Escape delimiter = ampersand [38]	

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

Order requests (and patient data)

P - Patient Record (Level 1)

Position in record	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
1	Char	1	Required	Identifier for Patient record (P)	
2	Char	1	Required	Sequence number. Single digit counting P records.	
4	ST	20	Required	Laboratory Patient ID. This must be an ASCII 7 string. Whitespace, extended ASCII or Unicode is not supported. It is possible to reconfigure this in cobas IT 3000 (Administration > Location > [Location:Name] > Configuration), so that it is read from any one of fields 3, 4 or 5.	<PID PATID>
6.1	ST	48 in total for 6	Required	Patient name: Surname - use component delimiter to separate all components in position (field) 6	<PID LASTNAME>
6.2	ST	48 in total for 6	Required	Patient name: First name - use component delimiter to separate all components in position (field) 6	<PID FIRSTNAME>
8	TS	14	Required	Date of birth (format: YYYYMMDD)	<PID DOB>
9	Char	1	Required	Sex: F=Female; W=Female; M=Male; U = Unknown. Any other value = Unknown	<PID SEX>

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

O- Order Record (Level 2)

Position in record	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
1	Char	1	Required	Identifier for Order record (O)	
2	Char	1	Required	Sequence number. Single digit counting consecutive O records.	
3	NM	22	Required	Order ID or Sample ID. 👁 For details on setting and using these values, see <i>Setting the sample ID and order number</i> on page A-45. It is possible to configure cobas IT 3000 to read either the sample ID or the Order number from field 4 instead, (Administration > Location > [Location_Name] > Configuration > ASTM_IN).	The first 10 digits to <ORC ORDERNR>. The whole number to <ORC SAMPLEID>
5.4	ST		Required	Host Test ID, i.e. Required test code. This is the local code that identifies the analyte.	<OBR TESTCODE>
6	ID	1	Optional	Priority. <ul style="list-style-type: none"> S = stat (short turn-around time). R = routine. Any other values defaults to "routine".	<ORC PRIORITY>
7	ID	1	Required	The date and time the order was sent in YYYYMMSSHHMMSS format.	<ORC ORDERDATE>

Position in record	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
8	ID	1	Optional	The date and time the sample was obtained in YYYYMMSSHHMMSS format.	<ORC WITHDRAWDATE>
12	ID	2	Required	Action code. In any one order, tests with the same test code (O-5.4) must all have the same action code. The supported values are: <ul style="list-style-type: none"> A=Add test; C=Delete existing test; N=Delete existing test; R=Delete test; G=Repeat test. <p>👁 For more details, see <i>Handling different actions in the order</i> on page C-27.</p>	<OBR ACTION>
16	ST		Optional	Code for the specimen type (indicating blood, urine etc.). (Note that version 2.03.08 and earlier of cobas IT 3000 placed this data in field 15.)	<ORC SPECIMEN>
17	ST		Optional	Code for the doctor or department that ordered the test.	<ORC ORDERER>
21	ST		Optional	Order status. Supported values are: <ul style="list-style-type: none"> A = Add test to order. CA = Sample was deleted (to cancel an order, all samples in the order must be deleted). SC = Order is in process, scheduled. R = Order has been replaced, or rerun order. RP = Order has been replaced, or rerun order. U = Update order. This recreates the order with specified tests, plus any previous tests that fulfil the conditions specified in the option HIS_UPDATE_DELETE_LEVEL. <p>👁 For details of HIS_UPDATE_DELETE_LEVEL, see <i>Update delete level</i> on page C-28.</p> <p>This value is used in conjunction with the Order Control ID, passed in O-26, and the Action Code, passed in O-12, to instruct cobas IT 3000 how to handle the order.</p> <p>👁 For more details, see <i>Handling different actions in the order</i> on page C-27.</p>	<ORC STATUS>
26	ST		Optional	Report type. Permitted values are: <ul style="list-style-type: none"> O = new order. X = cancel order, delete sample. C = change or delete previously sent order. <p>👁 For more details, see <i>Handling different actions in the order</i> on page C-27.</p>	<ORC CONTROL>

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

You cannot send orders containing results

Although the default templates contain fields for orders containing results, this is not supported in the current release. This may be supported in future releases. If you require this functionality, please contact Roche Diagnostics Global Customer support.

Order requests (and patient data)

L - Message Terminator Record (Level 0)

This is the last record, and identifies the end of the message.

Position in record	Data type ^(a)	Max length	Required/optional?	Description
1	Char	1		Identifier for Terminator record (L)
2	Char	1		Sequence number. A row counter. Always set to 1, as there is only ever one Terminator record.
3	ID	1		Code that gives an explanation of the reason for ending the message.

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

Values not mapped by default

There are further values that the host can send to **cobas IT 3000** in an order request, but are not transferred by the default XSL templates. To use these values, a service engineer will need to edit the XSL templates to support them.

👁 For details of how the XSL templates control the data passed between **cobas IT 3000** and the host, see *Transformation of an ASTM message* on page D-5.

Position in record	Data type ^(a)	Max length	Required/optional?	Description
-	ST	30	Optional	It is possible to use an alphanumeric alternative for the order number of sample number. In this case, the host should send the alphanumeric code identifying the order. Up to 30 characters are supported. This is suggested as a value for field O-3, component 1 "Specimen ID". <ORC HOST_ORDER_ID>
	ST		Required	The host's code identifying the physician dealing with the patient. This value is required if the other values are used. This is suggested as a value for the Patient record, field 14, component 1 "Attending physician code ID". <ORC PHYSICIAN_CODE>
	ST			The name of the physician. This is suggested as a value for Patient record, field 14, component 2 "Attending physician name". Note that currently, cobas IT 3000 does not support dividing the name into last name, first name, title, etc. <ORC PHYSICIAN_NAME>
	ST			A boolean value Y or N, telling cobas IT 3000 to add the physician's details to its database, if the details are not already there. The default setting is Y. To find the physician's preexisting details, cobas IT 3000 searches for a match using the physician's host code. If the code is not supplied, cobas IT 3000 searches by name for a match. <ORC PHYSICIAN_CREATE>

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

Example messages

In this example, at 16:20 on 10th January 2008, the host requests the **cobas IT 3000** for six measurements on a serum sample from Mary Nesbitt, born 4th April 1957:

```
H|\^&|200801101620544936
P|1|0001214173|0001214173|Nesbitt^Mary||19570404|F
O|1|500101999||102|||||A|||Serum|S0003|||
O|2|500101999||103|||||A|||Serum|S0003|||
O|3|500101999||106|||||A|||Serum|S0003|||
O|4|500101999||107|||||A|||Serum|S0003|||
O|5|500101999||108|||||A|||Serum|S0003|||
O|6|500101999||109|||||A|||Serum|S0003|||
L|1|N
```

Figure C-8 New order message for six tests on a serum sample

This shows a similar new order message. This message will successfully pass the order to **cobas IT 3000**, but the data in the unsupported fields will not be read.

```
H|\^&|||ASTM-Host||||PSM|P|20000219111500
P|1|923506|4637463G66||Smith^John||19630101|M||||||Diabetes|||20000218102000|||||Urology
O|1|923506|92350601|^^^33|R|20090907121223|20090907121223|||A|A|||||||O
O|2|923506|92350601|^^^36|R|20090907121223|20090907121223|||A|A|||||||O
O|3|923506|92350603|^^^1|R|20090907121223|20090907121223|||A|A|||||||O
L|1|F
```

Figure C-9 New order message

This shows a similar order message in batch mode.

```
H|\^&|||ASTM-Host||||PSM|P|20000219111500
P|1|923507|4637463XXX||Smith^John||19630101|M||||||Diabetes|||20000218102000|||||Urology
O|1|923507|923507|^^^260\^^^2\^^^1|R|20090907121223|20090907121223|||A|||||||O
L|1|F
```

Figure C-10 New order message in batch mode

This shows a Delete Test message. (C/A/R) This deletes tests without results. Tests with results, not released as well as released, will be kept.

```
H|\^&|||ASTM-Host||||PSM|P|20000219111500
P|1|923506|4637463G66||Smith^John||19630101|M||||||Diabetes|||20000218102000|||||Urology
O|1|923506|923506|^^^33|R|20090907121223|20090907121223|||R|||||A||||C
O|2|923506|923506|^^^36|R|20090907121223|20090907121223|||R|||||A||||C
L|1|F
```

Figure C-11 Delete Test of an Order (C/A/R)

This shows a Delete Test of an Order message (C/A/R) in batch mode. This deletes tests without results. Tests with results, not released as well as released, will be kept.

```
H|\^&|||ASTM-Host||||PSM|P|20000219111500
P|1|923506|4637463G66||Smith^John||19630101|M||||||Diabetes|||20000218102000|||||Urology
O|1|923506|923506|^^^33\^^^36\^^^1|R|20090907121223|20090907121223|||R|||||A||||C
L|1|F
```

Figure C-12 Delete Test of an Order (C/A/R) in batch mode

This shows a Delete sample message. To implement this, set SAMPLE_CREATE_RULE=1 (i.e. ORDERID != SAMPLEID) for testing of following dataset (samples with specimen 01 and 02).

Order requests (and patient data)

```
H|\^&|||ASTM-Host|||PSM|P|20000219111500
P|1|923506|4637463G66||Smith^John||19630101|M|||||Diabetes|||20000218102000|||||Urology
O|1|923506|92350601|^^^33|R|20090907121223|20090907121223||R|R|||||CA||||C
O|2|923506|92350602|^^^36|R|20090907121223|20090907121223||R|R|||||CA||||C
L|1|F
```

Figure C-13 Delete sample message

This shows an alternative Delete sample message. This example is valid, as long as you have different samples per message (and ORDERID != SAMPLEID).

```
H|\^&|||ASTM-Host|||PSM|P|20000219111500
P|1|923506|4637463G66||Smith^John||19630101|M|||||Diabetes|||20000218102000|||||Urology
O|1|923506|92350601|^^^R|20090907121223|20090907121223||R|R|||||X
O|2|923506|92350603|^^^R|20090907121223|20090907121223||R|R|||||X
L|1|F
```

Figure C-14 Delete sample message

This shows a Delete order message.

```
H|\^&|||ASTM-Host|||PSM|P|20111014111010
P|1|923506|4637463G66||Smith^John||19630101|M|||||Diabetes|||2011101411110|||||Urology
O|1|923506|97050307|^^^133|R|20111014111010|20111014111010||R|||01|WA1|||||X
L|1|F
```

Figure C-15 Delete order message

This shows a Change in patient ID (added, deleted or changed) message. To implement this, the option MODUL/HIS_ALLOW_CHANGE_ORDER must be switched on

```
H|\^&|||ASTM-Host|||PSM|P|20000219111500
P|1|923506|4637463XXX||Smith^John||19630101|M|||||Diabetes|||20000218102000|||||Urology
O|1|923506|923506|^^^33|R|20090907121223|20090907121223||R|R|||||SC||||C
O|2|923506|923506|^^^36|R|20090907121223|20090907121223||R|R|||||SC||||C
L|1|F
```

Figure C-16 Change in patient ID message

This shows a Repeat message. This overwrites results, even if they were already released.

```
H|\^&|||ASTM-Host|||PSM|P|20000219111500
P|1|923506|4637463XXX||Smith^John||19630101|M|||||Diabetes|||20000218102000|||||Urology
O|1|923506|923506|^^^33|R|20090907121223|20090907121223||G|G|||||RP||||C
O|2|923506|923506|^^^36|R|20090907121223|20090907121223||G|G|||||R||||C
L|1|F
```

Figure C-17 Repeat message

This shows a Repeat message in batch. This overwrites results, even if they were already released.

```
H|\^&|||ASTM-Host|||PSM|P|20000219111500
P|1|923506|4637463XXX||Smith^John||19630101|M|||||Diabetes|||20000218102000|||||Urology
O|1|923506|923506|^^^33\^^^36\^^^1|R|20090907121223|20090907121223||G|G|||||RP||||C
L|1|F
```

Figure C-18 Repeat message in batch

This shows a Delete all tests from order message. This deletes tests independently of their result status (also deleted if result already released). It adds the last test of the message (here it would delete all but add test with code 36 only).

```
H|\^&|||ASTM-Host|||PSM|P|20000219111500
P|1|923506|4637463XXX||Smith^John||19630101|M|||||Diabetes|||20000218102000|||||Urology
O|1|923506|923506|^^^33|R|20090907121223|20090907121223|||R|R|||||C
O|2|923506|923506|^^^36|R|20090907121223|20090907121223|||R|R|||||C
L|1|F
```

Figure C-19 Delete all tests from order

This shows an alternative Delete all tests from order message. This deletes tests independently of their result status (also deleted if result already released) and it adds all the tests which are part of the message.

```
H|\^&|||ASTM-Host|||PSM|P|20000219111500
P|1|923506|4637463XXX||Smith^John||19630101|M|||||Diabetes|||20000218102000|||||Urology
O|1|923506|923506|^^^33\^^^36\^^^1|R|20090907121223|20090907121223|||R|R|||||C
L|1|F
```

Figure C-20 Delete all tests from order

This shows an update order message. This deletes tests that have not yet returned results, and adds all the tests which are part of the message.

```
H|\^&|||ASTM-Host|||PSM|P|20120322162330
P|1|923506|4637463XXX||Smith^John||19630101|M|||||Diabetes|||20120322140130|||||Urology
O|1|923506|97050114|^^^133\^^^134|R|20120322140130|20120322140145|||A|||||U||||C
L|1|F
```

Figure C-21 Update order

Handling different actions in the order

This table explains how to instruct cobas IT 3000 to perform actions on the order, using the fields in the Order record.

	Field O-12 (Action code)	Field O-21 (Order status)	Field O-26 (Report type)
Add test	A		O
Delete test	R	A	C
Delete order (if no sampleID is sent, complete order will be deleted; if sampleID is sent, only the sample will be deleted)			X
Rebuild new sample (delete all tests for the sample and add the new ones)	R		C
Delete sample (delete all tests for the sample and delete the sample itself)	R	CA	C
Delete sample (as above)	R		X
Change in PID	R	SC	C
Rerun / reflex	G	RP	C
Update order	A	U	C

Update delete level

When an update order message is sent, the order is recreated with the new tests specified. The order's previous tests are deleted, except those that meet the criteria specified in the parameter HIS_UPDATE_DELETE_LEVEL. This parameter is set in **cobas IT 3000** in Administration > Options > MODUL > HIS_UPDATE_DELETE_LEVEL.

The parameter HIS_UPDATE_DELETE_LEVEL is a bitwise binary flag. It consists of an 8-digit binary number, each digit of which can be either 1 or 0. Each digit sets a different flag, as detailed in the following chart.

Do not delete if test:	Binary flag	Code
(Never delete)	10000000	DEL_LVL_NEVER
Has been transferred	01000000	DEL_LVL_TRANSFERRED
Has been scanned	00100000	DEL_LVL_SCANNED
Has been put on work list	00010000	DEL_LVL_WORK_LIST
Has been seen	00001000	DEL_LVL_SEEN
Has been processed by SDI	00000100	DEL_LVL_DONE_BY_SDI
Has a result	00000010	DEL_LVL_RESULT
Has a released result	00000001	DEL_LVL_RELEASED_RESULT

As a binary flag, these values can be combined. For example the setting 00011011 tells **cobas IT 3000** not to delete any pre-existing tests that meet one or more of the following criteria:

- tests that have been added to the worklist (00010000)
- tests that have been seen by **cobas IT 3000** and sent to the instrument (00001000)
- tests that have a result (00000010)
- tests that have a released result (00000001)

The default value of HIS_UPDATE_DELETE_LEVEL is 00011011.

Automatic creation of orderer

If using the default templates, you must pre-program **cobas IT 3000** with the orderers that will send orders. If you send an orderer which **cobas IT 3000** does not know of, it cannot return test results. The default templates read the orderer information from the ASTM Order record, field 17, and assign it to ORC/@ORDERER.

However, you can configure **cobas IT 3000** to automatically create a new orderer, whenever it receives an order from a new source. In this case, you must configure **cobas IT 3000** to read the orderer information from the ORDERER element. The ASTM Order Request message must be transformed into an XML document similar to the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<OrderRequest>
  <MSH SENDER="TEST"/>
  <PID PATID="TH" LASTNAME="HARRY" FIRSTNAME="TEST" DOB="19010101" SEX="M">
    <ORC ORDERNR_ORG="1453376" ORDERNR="453376" SAMPLEID="1453376" SPECIMEN="" PRIORITY="R" STA-
TUS="" CONTROL="NW">
      <ORDERER DEFAULT_AUFGCD="?" HOSTCODE="WARD1" NAME="WARD1" WARDID="WARD1" WARD="J"/>
      <OBR TESTCODE="NA" ACTION="A"/>
      <OBR TESTCODE="K" ACTION="A"/>
      <OBR TESTCODE="CREAT" ACTION="A"/>
      <OBR TESTCODE="MG" ACTION="A"/>
    </ORC>
  </PID>
</OrderRequest>
```

Figure C-22 Example Order Request message for automatic creation of orderer

To implement this, you must edit the XSLT stylesheets for the Order Request.

👁 For details of how to find and edit the XSLT stylesheets, see *Making extensive reconfigurations* on page A-15.



How to edit XSLT

To edit the XSLT stylesheets, you should to learn at least some basic XSLT and XPath. You can find tutorials and reference information on XSLT and XPath on the internet, for example at <http://www.w3schools.com>. Please note that Roche Diagnostics is not responsible for the content of this or any other external website.

► To implement automatic creation of orderer in ASTM

- 1 Confirm in what ASTM field the host sends the orderer (or sender) information. Typically this will be in O-17, but some sites send it in P-26 or another field.
- 2 Navigate to **Administration > Client (or Location) > [Name of Location] > Configuration**, and double click the template for order requests, by default **OrderRequest**. A dialog box opens, containing the Order Request XSLT.
- 3 Copy and paste all the content of the dialog box to your preferred XML editor (such as Stylus Studio or Altova XML Spy), or to your preferred text editor (such as Notepad or Notepad++). Make a backup copy, and then edit the file in your chosen editor.
- 4 Find and delete the section in the template that assigns a value to the attribute "ORDERER".
- 5 Find the definition of the last attribute of the ORC element. The ORC element starts with the tag:

```
<xsl:element name="ORC">
```

It ends with

```
</xsl:element>
```

Inside this element there are several attributes, defined with tags:

```
<xsl:attribute ...> ... </xsl:attribute>
```

There is also a sub-element, OBR, defined with the tags:

```
<xsl:element name="OBR"> ... </xsl:element>
```

Put your new commands between the last `</xsl:attribute>` tag and the `<xsl:element name="OBR">` that starts the OBR element.

- 6 Enter the XSLT commands to assign the orderer information to the ORDERER element. For example, if the orderer information is sent in ASTM field O-17, add, as the first sub-element in the ORC element:

```
<xsl:for-each select="FIELD[@NO='17']/DATA/COMP[@NO='1']">
  <xsl:element name="ORDERER">
    <xsl:attribute name="DEFAULT_AUFGCD"?></xsl:attribute>
    <xsl:attribute name="HOSTCODE"><xsl:value-of select="@TEXT"/></xsl:attribute>
    <xsl:attribute name="NAME"><xsl:value-of select="@TEXT"/></xsl:attribute>
    <xsl:attribute name="WARDID"><xsl:value-of select="@TEXT"/></xsl:attribute>
    <xsl:attribute name="WARD">J</xsl:attribute>
  </xsl:element>
</xsl:for-each>
```

Figure C-23 Example XSLT to read automatic orderer creation data from ASTM O-17.

If the orderer information is sent in another ASTM field, edit this example to meet your specific needs. For example if the orderer information is sent in P-26, the first line of this code should read:

```
<xsl:for-each select="../FIELD[@NO='26']/DATA/COMP[@NO='1']">
```

- 7 Save your changes in the editor, paste the contents of the edited file back into **cobas IT 3000**, and save your changes in **cobas IT 3000**.
- 8 Make a note of the default orderer you defined in the XSLT, in the DEFAULT_AUFGCD element ("?" in this example). Make sure this orderer is defined in **Workplaces > Parameter > Orderer**. Every orderer field not defined in the orderer element will be taken from the default orderer.
- 9 Test your host interface thoroughly before implementing it live.

The fields available in the ORDERER element

The ORDERER element is not used by the default **cobas IT 3000** templates. By editing the templates, the following fields are available. For more details on these fields see the online help for the dialog in **Workplaces > Parameter > Orderer > Insert new orderer**. Some fields, for example the used report configuration, can not be configured via the host interface.

HL7 Segment	Pos. in HL7	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
-	-	ST		Optional	The name of the default orderer. The default orderer must exist in Workplaces > Parameter > Orderer . Every orderer field not defined in the orderer element will be taken from the default orderer.	<ORDERER DEFAULT_AUFGCD>
-	-	ST	5	Required	The code used in the host for the orderer.	<ORDERER HOSTCODE>
-	-	ST		Required	The name of the orderer.	<ORDERER NAME>
-	-	ST		Optional	Text description of the orderer, to help a user with identification in reports.	<ORDERER REMARK>
-	-	ST		Optional	The cost center associated with the orderer.	<ORDERER COSTCENTER>
-	-	ST		Optional	The code that determines the order in which orderers appear on reports.	<ORDERER SORT>
-	-	ST		Optional		<ORDERER TITLE1>
-	-	ST		Optional		<ORDERER TITLE2>
-	-	ST		Optional		<ORDERER ZIP>
-	-	ST		Optional		<ORDERER CITY>
-	-	ST		Optional		<ORDERER BILLINGCODE>
-	-	ST		Optional	A pseudo-boolean value. Optional, but must be set to “J” in production environments, or else results are not returned.	<ORDERER WARD>
-	-	ST		Optional	(This functionality is not currently supported.)	<ORDERER TOUR>
-	-	ST		Optional	The orderer group defined in cobas IT 3000 , in which to place the new orderer.	<ORDERER ORDERERGROUP>
-	-	ST		Optional	The organization to associate the orderer with in cobas IT 3000	<ORDERER ORGANIZATION>
-	-	ST		Optional	The code to give to the orderer in cobas IT 3000 .	<ORDERER ABBREVIATION>
-	-	ST		Optional		<ORDERER PHONE>
-	-	ST		Optional		<ORDERER FAX>
-	-	ST		Required	The ward ID for the new orderer, used in ward communication.	<ORDERER WARDID>

Automatic creation of orderer

- (a) Data type abbreviations: ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for HL7-defined tables

Test results

Roche Diagnostics **cobas IT 3000** sends this message to the laboratory information system (LIS). This message gives the results of the tests on the specimens.

Message structure

A **result message** contains the following segments:

Segment	Segment name	Description
H	Message header record	General information about message, including the message type
P	Patient record	Patient information such as ID, name and first name, etc. (incl. information about the visit)
O	Order record	General information about order and the requested tests
R	Result record	Test result
C	Comment record	Comments about the result and its interpretation
L	Terminator record	Ends the message.

Table C-5 Structure of an observation result

Configuring the interface with parameters

You can configure the host interface by using the parameters in **Workplaces > Administration > Clients > [your client] > Configuration**. The default stylesheet provides the following for result messages sent from **cobas IT 3000**:

Parameter	Type	Default value	Other possible values	Instruction to cobas IT 3000:
transferComments	Boolean	false()	true()	If set, the NTE comment segments are included in the result message.
test	Boolean	false()	true()	Set this field to "false()" in production environments.

Table C-6 Parameters used for result messages.

Integer and string values must be in single quotation marks. Otherwise Oracle may behave inconsistently.

- 👁 For details of how the XSL templates control the data passed between **cobas IT 3000** and the host, see *Transformation of an ASTM message* on page D-5.
- 👁 For details of creating a parameter, see *Creating a parameter* on page A-17.

Field mappings

This list contains the ASTM fields generated by **cobas IT 3000** and provides a short description for each field.

Test results

H - Message Header Record (Level 0)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier for Header record (H)	
2			Delimiter definition. The following standard delimiters are recommended: \ ^ &	
	Char	1	: Field delimiter = vertical bar [124]	
	Char	1	\: Repeat delimiter = backslash [92]	
	Char	1	^: Component delimiter = caret [94]	
	Char	1	&: Escape delimiter = ampersand [38]	
5	ST		Sending application	<MSH SENDING_APPLICATION>
10	ST		Receiving application	<MSH RECEIVING_APPLICATION>
12	Char	1	Processing ID, which indicates how the message is to be processed. In this case, P = "Production".	
14	ST		Date message generated, in YYYYMMDDHHMMSS format.	<MSH DATE>

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

P - Message Patient Record (Level 1)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier that starts a Patient record (P)	
2	NM		Identifier for Patient records. Set to 1 for the first patient record in the message, and then increments by one for each subsequent record. In the current version (2.04.00) only one patient record is supported.	<PID SET>
3	ST		The sample identifier. Note that the ASTM definitions put a practice-defined identifier for the patient in this field.	<OBR SAMPLEID>
6.1	ST		The patient's surname.	<PID LASTNAME>
6.2	ST		The patient's first name.	<PID FIRSTNAME>
8	ST		The patient's date of birth in YYYYMMDD format.	<PID DOB>
9	Char		The patient's sex. Takes either M, F, or U.	<PID SEX>

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

O - Message Order Record (Level 2)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier that starts an Order record (O)	
2	NM		Identifier for Order records. Set to 1 for the first order record per patient, and the increments by 1 for each subsequent order for that patient.	<ORC SET>
3	ST		Identifier for the sample or specimen.	<ORC SAMPLEID>
4.2	ST		The tray number, of the sample, as the first part of the Instrument Specimen ID.	<OBX TRAY>
4.3	ST		The position of the sample on the tray, as the second part of the Instrument Specimen ID.	<OBX POSITION>
5	ST		The Universal Test ID. Set to ALL meaning that all the associated results are following in the result records of the message.	
7	ST		The date and time the test was ordered, in YYYYMMDDHHMMSS format.	<ORC REQUESTDATE>
26	Char		Report Type. In this case, F = final results.	

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

R - Message Result Record (Level 2)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier that starts a Result record (R)	
2	NM		Identifier for Result records. Set to 1 for the first result record for the patient, and increments by one for each further result for the same patient.	<OBX SET>
3.4	ST		Test Code. (Manufacturer's local code in Universal Test ID).	<OBX ID>
4	ST		Data or measurement value.	<OBX VALUE>
5	ST		Units of measurement. (ISO 2955).	<OBX UNITS>
9	Char		Field that indicates the status of the result. In this case, F = final results.	

Test results

12	ST	The instrument timestamp. This is considered to be the time the test started. <OBX INSTRUMENT_MEASUREMENT_TIME>
13	ST	Date and time that cobas IT 3000 received the test result, considered to be the time the test was completed. <OBX COMPLETEDATE>
22	Char	The confidentiality of the test result. This is set for the test, by a parameter in cobas IT 3000 : <ul style="list-style-type: none"> C = confidential test N = normal test <OBX CONFIDENTIAL>

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

C - Message Comment Record (Level 2)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier that starts a Comment record (C).	
2	NM		Identifier for Comment records. Set to 1 for the first comment record for the result, and increments by one for each further comment for the same result.	<NTE SET>
4	ST		Text of the comment.	<NTE TEXT>

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

L - Message Terminator Record (Level 0)

Position in record	Data type ^(a)	Max length	Description
1	Char	1	Identifier that starts a Terminator record (L)
2	Char	1	This is always set to 1.
3	Char	1	The record termination code. Set to N = normal termination.

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

Sample message This example shows the result of Mary Nesbitt's tests that **cobas IT 3000** returns to the host:

```

H|\^&|||ROCHE|||PSM||P||20090511145230|
P|1|500101999||Nesbitt^Mary||19570404|W|
O|1|500101999|^^^^^|ALL||20090511121008|F|
R|1|^^^102|23|U/L|||F|||20090511145204|20090511145213|
R|2|^^^103|34|U/L|||F|||20090511145207|20090511145216|
R|3|^^^106|45|umol/L|||F|||20090511145208|20090511145218|
R|4|^^^107|23,00|mmol/L|||F|||20090511121900|20090511121908|
R|5|^^^108|24|mmol/L|||F|||20090511121901|20090511121911|
R|6|^^^109|25,0|mmol/L|||F|||20090511121905|20090511121915|
L|1|N|

```

Figure C-24 Results of tests returned to host

Values not mapped by default

There are further values that **cobas IT 3000** can send to the host in test result message, but are not transferred by the default XSL templates. If you wish to use these values, you will need to edit the XSL templates to support them.

👁 For details of how the XSL templates control the data passed between **cobas IT 3000** and the host, see *Reading the ASTM or HL7 messages* on page A-10.

	Data type ^(a)	Max length	Description
-	ST	30	It is possible to use an alphanumeric alternative for the order number of sample number. In this case, the host should send the alphanumeric code identifying the order. Up to 30 characters are supported. This is suggested as a value for field O-3, component 1 “Specimen ID”. <div><ORC HOST_ORDER_ID></div>
	ST		The host’s code identifying the physician dealing with the patient. This value is required if the other values are used. This is suggested as a value for the Patient record, field 14, component 1 “Attending physician code ID”. <div><ORC PHYSICIAN_CODE></div>
	ST		A boolean value Y or N, telling cobas IT 3000 to add the physician’s details to its database, if the details are not already there. The default setting is Y. To find the physician’s preexisting details, cobas IT 3000 searches for a match using the physician’s host code. If the code is not supplied, cobas IT 3000 searches by name for a match. <div><ORC PHYSICIAN_CREATE></div>
	ST		The name of the physician. This is suggested as a value for Patient record, field 14, component 2 “Attending physician name”. Note that currently, cobas IT 3000 does not support dividing the name into last name, first name, title, etc. <div><ORC PHYSICIAN_NAME></div>
-	ST		The dilution factor for result. <div><OBX VERDFAKT></div>
-	NM		A numeric boolean flag to say if there is a comment for the result: <ul style="list-style-type: none"> 0 = no comment 1 = result comment <div><OBX COMMENT></div>

Table C-7 Values accepted by **cobas IT 3000**, but not mapped by default, in a test result message

Test results

	Data type ^(a)	Max length	Description	
-	ST		This is a number that distinguishes an order from other orders sent on the same day. For each order sent during the course of a day, this number increments by 1. Every day at midnight the number is reset to zero. The number cannot be set manually, but increments up to a maximum of 999999 per day.	<OBX DAYNUMBER>
-			A graphical display of result compared to the normal range defined for the analyte.	<OBX GRAPH>
-	NM	10	The code for the analyte used by the host system. This is the value in the field of Administration > KIS > [LISDB interface] > Analyte assignment > External Analyte (Out) .	<OBX HOSTCODE>
-	ST		The Location that handles the message, as defined in Administration > Location > [Location Name] or Administration > Client > [Client Name] .	<OBX LOCATION>
-	ST		The name of the analyte.	<OBX NAME>
-	ST		The code for a test profile.	<OBX PROFILE>
-	ST		The abbreviated analyte synonym, as shown in the field in Parameter > analytes/ref. ranges > Abbr.	<OBX SYNONYM>

Table C-7 Values accepted by **cobas IT 3000**, but not mapped by default, in a test result message

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

Quality control results

Roche Diagnostics **cobas IT 3000** sends quality control messages to the laboratory information system (LIS). These messages give the results of the tests on the quality control samples.

Message structure

An **Quality control message** contains the following segments:

Segment	Segment name	Description
H	Message header record	General information about message, including the message type
P	Patient record	Patient information such as ID, name and first name, etc. (incl. information about the visit)
O	Order record	General information about order and the requested tests
R	Result record	Test result
L	Terminator record	Ends the message

Table C-8 Structure of a QC result message

Configuring the interface with parameters

There are no parameters for configuring ASTM quality control results.

Field mappings

This list contains the ASTM fields generated by **cobas IT 3000** and provides a short description for each field.

H - Message Header Record (Level 0)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier for Header record (H)	
2			Delimiter definition. The following standard delimiters are recommended: ^ &	
	Char	1	: Field delimiter = vertical bar [124]	
	Char	1	\: Repeat delimiter = backslash [92]	
	Char	1	^: Component delimiter = caret [94]	
	Char	1	&: Escape delimiter = ampersand [38]	
5	ST		Sending application	<MSH SENDING_APPLICATION>
10	ST		Receiving application	<MSH RECEIVING_APPLICATION>
12	Char	1	Processing ID. Set to P = Production.	
14	ST		Date message generated, in YYYYMMDDHHMMSS format.	<MSH DATE>

Quality control results

- (a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

P - Message Patient Record (Level 1)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier that starts a Patient record (P)	
2	NM		Identifier for Patient records. Set to 1.	<PID SET>

- (a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

O - Message Order Record (Level 2)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier that starts an Order record (O)	
2	NM		Identifier for Order records. In this case, set to 1.	
3	ST		The control ID. String identifying the control.	<ORC CONTROLID>
26	Char		Report type. In this case, F = Final results.	

- (a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

R - Message Result Record (Level 2)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier that starts a Result record (R)	
2	NM		Identifier for Result records. Set to 1 for the first result record for the same test, and increments by one for each further result for the same test.	<OBX SET>
3.4	ST		Test Code. This is the external test code, i.e. the host's code, for the analyte, for example 102, 103, 106 etc., or K, NA, CL etc., depending values set.	<OBR ID>
3.5	ST		Test Code. This is the external test code, i.e. the host's code, for the analyte, for example 102, 103, 106 etc.	<OBX ANALYTNR>
4	ST		Data or measurement value.	<OBX VALUE>
5	ST		Units of measurement. (ISO 2955).	<OBX UNITS>
7	Char		The normalcy of the result. In this case, set to N = normal.	

12	ST	The instrument timestamp. This is considered to be the time the test started. <OBX INSTRUMENT_MEASUREMENT_TIME>
13	ST	Date and time cobas IT 5000 completed the test, in YYYYMMDDHHMMSS format. Note that in version 2.03.08 and earlier of cobas IT 3000 , this value was sent in field 12. <OBX COMPLETEDATE>
14	ST	Identifier for the instrument that performed the test. <OBR INSTRUMENT>

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

L - Message Terminator Record (Level 0)

This is the last record, and identifies the end of the message.

Position in record	Data type ^(a)	Max length	Description
1	Char	1	Identifier for a Terminator record (L)
2	Char	1	A row counter. Always set to 1, as there is only ever one Terminator record.
3	ID	1	Code that gives an explanation of the reason for ending the message. This is hard-coded to N for Normal.

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

Sample message This example shows the cholesterol test result on a low-normal sample that **cobas IT 3000** returns to the host:

```
H|^&~\|||P242|||GWI-LIS||Q||20080110171838<CR>
P|1|<CR>
O|1|||F<CR>
R|1|^HDL^132|1.0|mmol/l|||F|||200704165021|BOEHLIM 20.04.07<CR>
R|2|^LDL^133|2.60|mmol/l|||F|||200704165021|BOEHLIM 20.04.07<CR>
L|1|N
```

Figure C-25 Example quality control result message.

Values not mapped by default

There are further values that **cobas IT 3000** can send to the host with a quality control result, but are not transferred by the default XSL templates. If you wish to use these values, you will need to edit the XSL templates to support them.

👁 For details of how the XSL templates control the data passed between **cobas IT 3000** and the host, see *Reading the ASTM or HL7 messages* on page A-10.

Position	Data type ^(a)	Max length	Required/optional?	Description	XML tag/Attribute
-	ST		Optional	The name of the control, for example PNU or PPU. This is a suggested value for the Order record, field 3, component 1, instead of the patient sample ID.	<ORC CONTROL_NAME>
-	NM		Optional	The lot number of the control material. This is a suggested value for the Order record, field 3, component 2, instead of the patient sample ID.	<ORC CONTROL_LOT>
-	IS		Optional	The ID number used by the host to identify the control. This is a suggested value for the Order record, field 3, component 3, instead of the patient sample ID.	<ORC CONTROL_HOST>
-	IS		Optional	An alternative ID number used by the host to identify the control. This is a suggested value for the Order record, field 3, component 3, instead of the patient sample ID.	<ORC CONTROL_HOST_ID>
-	ST		Optional	This is ID for the instrument and the module that performed the quality control test. This is suggested as a value for the Result record, field 14.	<OBX GERNAME>

(a) Data type abbreviations: ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for HL7-defined tables

Sample Event message

Roche Diagnostics **cobas IT 3000** sends this message to the laboratory information system (LIS). This informs the HIS/LIS about the location of a specimen, but without making a query for orders.

Message structure

A **Sample Event message** contains the following segments:

Segment	Segment name	Description
H	Message header record	General information about message, including the message type
[
P	Patient record	Patient information such as ID, name and first name, etc. (incl. information about the visit)
O	Order record	General information about order and the requested tests
]		
M 1 EQU	Equipment detail record	Data necessary to identify and maintain the equipment that is being used throughout the Laboratory Information System, and to notify the host or Hospital Information System of the status of the sample.
M 2 SAC	Container detail record	Data necessary to maintain the containers, such as sample tubes, that are being used throughout the Laboratory Information System.
L	Terminator record	Ends the message.

Table C-9 Structure of Sample Event message

Configuring the interface with parameters

You can configure the host interface by using the parameters in **Workplaces > Administration > Clients > [your client] > Configuration**. The default stylesheet provides the following for messages with sample event messages from **cobas IT 3000**:

Parameter	Type	Default value	Other possible values	Instruction to cobas IT 3000:
ASTM_20	Boolean	true()	false()	Send the Patient and Order records in the message.
SENDING_APPLICATION	String		(any)	The value to write in MSH-3, for the “sending application”. MSH-3. If set, this value overrides any value configured in cobas IT 3000 .
SENDING_FACILITY	String		(any)	The value to write in MSH-4, for the “sending facility”. If set, this value overrides any value configured in cobas IT 3000 .
RECEIVING_APPLICATION	String		(any)	The value to write in MSH-5, for the “receiving application”. If set, this value overrides any value configured in cobas IT 3000 .
RECEIVING_FACILITY	String		(any)	The value to write in MSH-6, for the “receiving facility”. If set, this value overrides any value configured in cobas IT 3000 .

Table C-10 Parameters used for incoming messages.

Sample Event message

Parameter	Type	Default value	Other possible values	Instruction to cobas IT 3000:
SEND_ONCE	String			Not used in default template
SEND_INSTRUMENTS	String	ALL		Not used in default template
SEND_LOCATION	String	ALL		Not used in default template

Table C-10 Parameters used for incoming messages.

Integer and string values must be in single quotation marks. Otherwise Oracle may behave inconsistently.

Field mappings

This list contains the ASTM fields generated by **cobas IT 3000** and provides a short description for each field.

H - Message Header Record (Level 0)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier for Header record (H)	
2			Delimiter definition. The following standard delimiters are recommended: \ ^ &	
	Char	1	: Field delimiter = vertical bar [124]	
	Char	1	\: Repeat delimiter = backslash [92]	
	Char	1	^: Component delimiter = caret [94]	
	Char	1	&: Escape delimiter = ampersand [38]	
5.1	ST		Sending application. It is possible to configure the value of this field in the setting: Administration > Location > [Location_Name] > Configuration > SampleSeen_Astm. If nothing is specified, cobas IT 3000 generates an automatic value.	<MSH SENDING_APPLICATION>
5.2	ST		Sending facility. It is possible to configure the value of this field in the setting: Administration > Location > [Location_Name] > Configuration > SampleSeen_Astm. If nothing is specified, cobas IT 3000 generates an automatic value.	<MSH SENDING_FACILITY>
5.3	ST		The receiving application. It is possible to configure the value of this field in the setting: Administration > Location > [Location_Name] > Configuration > SampleSeen_Astm. If nothing is specified, cobas IT 3000 generates an automatic value.	<MSH RECEIVING_APPLICATION>
5.4	ST		The receiving facility. It is possible to configure the value of this field in the setting: Administration > Location > [Location_Name] > Configuration > SampleSeen_Astm. If nothing is specified, cobas IT 3000 generates an automatic value.	<MSH RECEIVING_FACILITY>
12	Char	1	Processing ID, which indicates how the message is to be processed.	<MSH PROCESSING_ID>
14	ST		Date message generated, in YYYYMMDDHHMMSS format.	<MSH EVENT_DATE_TIME>

- (a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

P - Message Patient Record (Level 1)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier that starts a Patient record (P)	
2	NM		Identifier for Patient records. Set to 1 for the first patient record in the message, and then increments by one for each subsequent record.	<PID SET>
3	ST		Practice-defined Identifier for the patient. This must be an ASCII 7 string. Extended ASCII or Unicode is not supported.	<PID EXTERNAL_PATIENT_ID>
4	ST		Laboratory-defined Identifier for the patient. This must be an ASCII 7 string. Extended ASCII or Unicode is not supported.	<PID PATIENT_ID>
6.1	ST		The patient's surname.	<PID LAST_NAME>
6.2	ST		The patient's first name.	<PID FIRST_NAME>
8	ST		The patient's date of birth in YYYYMMDD format.	<PID DATE_OF_BIRTH>
9	Char		The patient's sex. Takes either M, F, or U.	<PID SEX>

- (a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

O - Message Order Record (Level 2)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	Char	1	Identifier that starts an Order record (O)	
2	NM		Identifier for Order records. Set to 1 for the first order record per patient, and the increments by 1 for each subsequent order for that patient.	
3	ST		Identifier for the sample or specimen.	<SAC CONTAINER_IDENTIFIER>
8	ST		The date and time the specimen was collected, in YYYYMMDDHHMMSS format. This is presumed to be the same time as that specified in O-15 below.	<PID ORDER_DATE>
12	ST		The Action Code, indicating action to be taken. Set to x indicating that the specimen or test is already in process.	

Sample Event message

15	Char	The date and time the specimen was received, in YYYYMMDDHHMMSS format. <PID ORDER_DATE>
16	Char	Specimen Type. Descriptor of the specimen, for example SE or 01 for Serum. <SAC SPECIMEN_SOURCE>
26	Char	Report Type. In this case, I = in instrument, pending.

- (a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

M|1|EQU - Equipment Detail Record (Level 3)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	M	1	Identifier that starts an Manufacturer-defined record (M)	
2	NM		Identifier for the Manufacturer-defined record. In the case of the Equipment Detail Record, this is set to 1, to signify the first Manufacturer-defined record.	
3	ST		Identifier for the Equipment Detail Record. Set to EQU .	
4.4	ST		The name of the instrument. In the case of an archive message, this field shows instead the user logged into cobas IT 3000 . (Optional).	<EQU EQUIPMENT_IDENTIFIER_NAME>
4.5	ST		Location, as defined in cobas IT 3000 , from which the sample came, such as the hospital or laboratory. (Optional).	<EQU EQUIPMENT_IDENTIFIER_CLIENT>

4.6	ST	The function call. Set to SAMPLEEVENT. <EQU EQUIPMENT_IDENTIFIER_MODULE>
4.7	ST	The content of the message, identifying the event. This parameter is optional. Permitted values are: <ul style="list-style-type: none"> SEEN This identifies a Sample viewed message. DISPOSE This identifies a Sample disposed of message. LOESCH This identifies a Sample deleted message. NEU This identifies a Sample entered message. RESULT This identifies a Sample analyzed message. RETRIEVE This identifies a Sample retrieved message. SCAN This identifies a Sample scanned message. ARCHIV This identifies a Sample archived message. <EQU EQUIPMENT_IDENTIFIER_EVENT>
5	ST	This field is the date and time that the event (e.g., state transition, issuing of command, finishing of command execution) occurred. (Format: YYYYMMDDHHMMSS). <EQU EVENT_DATE_TIME>

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

M|1|SAC - Container Detail Record (Level 3)

Position in record	Data type ^(a)	Max length	Description	XML tag/Attribute
1	M	1	Identifier that starts an Manufacturer-defined record (M)	
2	NM		Identifier for the Manufacturer-defined record. In the case of the Container Detail Record, this is set to 2, to signify the second Manufacturer-defined record.	
3	ST		Identifier for the Container Detail Record. Set to SAC.	
6	ST		Sample id. Identifier for the sample or specimen. For an aliquot, this contains the identifier for the aliquoted sample or specimen. <SAC CONTAINER_IDENTIFIER>	
7	ST		Parent sample id. For an aliquot, the identifier for the parent sample or specimen from which the aliquot was made. In other cases, this field is empty. <SAC PRIMARY_CONTAINER_IDENTIFIER>	
9	Char		Specimen Type. Descriptor of the specimen, for example SE or 01 for Serum. <SAC SPECIMEN_SOURCE>	
10	TS		This field is the date and time that the container was last registered with the Laboratory Information System, e.g., reading of a container bar code by a device. (Format: YYYYMMDDHHMMSS). <SAC REGISTRATION_DATE_TIME>	

Sample Event message

11.1	ID	<p>This field contains a single character HL7 code that identifies the status of the unique container in which the specimen resides at the time that the transaction was initiated.</p> <ul style="list-style-type: none"> I Identified. The container has been received. P In Position. The container is in position for specimen transfer (e.g., container removal from track, pipetting, etc.). O In Process. The container is being processed by the equipment. It is useful as a response to a query about Container Status, when the specific step of the process is not relevant. R Process Completed. Processing has been completed, but the container has not been released. L Left Equipment. The container has been released from that system. M Missing. The container did not arrive at its next expected location. X Container Unavailable. The container is no longer available within the scope of the system (e.g., tube broken or discarded). U Unknown. The container has not been identified.
<SAC CONTAINER_STATUS>		
11.4	ST	The error code returned by the instrument, if applicable.
<SAC SPECIFIC_CONTAINER_STATUS>		
12	ST	The carrier type, for example the size of the tray e.g. 20x5.
<SAC CARRIER_TYPE>		
13	ST	The tray number, of the sample, as the first part of the Instrument Specimen ID.
<SAC CARRIER_IDENTIFIER>		
14	ST	The position of the sample on the tray, as the second part of the Instrument Specimen ID. If the tray is configured by column and row, this field shows the column. Otherwise, it shows the number that identifies the position of the sample.
<SAC POSITION_IN:CARRIER>		

(a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

L - Message Terminator Record (Level 0)

This is the last record, and identifies the end of the message.

Position in record	Data type ^(a)	Max length	Description
1	Char	1	Identifier for a Terminator record (L)
2	Char	1	A row counter. Always set to 1, as there is only ever one Terminator record.
3	ID	1	Code that gives an explanation of the reason for ending the message. This value is currently not read, but is set to N for "Normal termination".

- (a) Data type abbreviations: Char=Single character; ST=Alphanumeric string; NM=number; TS=Time stamp; IS=Coded value for user-defined tables; ID=Coded value for ASTM-defined tables

Sample message This example shows an archive message that **cobas IT 3000** returns to the host:

```
H|\^&|||CITSOLUTION^RD^IT3000^2.05.01|||||P||20000215094700
O|1|100000025001|||R|20080620122509|||X|||20080620124700|01|||||||I
M|1|EQU|^RSD800A^80^SAMPLEEVENT^ARCHIVE|20000215094700
M|2|SAC|||100000025001|100000025001||01|20000215094700|P|ARCHIVE|ARCHIVE-1|1|
L|1|N
```

Figure C-26 Sample Event, archive message.

This example shows a sample seen message that **cobas IT 3000** returns to the host:

```
H^~\&|||ROCHE^ROCHE-LAB^IT3000^2.05.01|||||
P|1||10774373||Brösel^Rainer||19871122|M|
O|1|100000103|||20090511|||X|||20090511|01|||||||I|
M|1|EQU|^EMBLA^LAB1^SAMPLEEVENT^SEEN|20090518084751|
M|2|SAC|||100000103|||01||P^^^|cobas 6000|0230|1^|
L|1|N
```

Figure C-27 Sample Event, sample seen message.

Sample Event message

Appendix

D

10	<i>Example XSLT transformations</i>	D-3
11	<i>The default XSLT transformations</i>	D-15
12	<i>XML schema definitions</i>	D-117

Example XSLT transformations

How files are transformed from ASTM or HL7 to XML

This chapter gives examples of how **cobas IT 3000** transforms ASTM and HL7 files into its internal XML formats.

In this chapter	Chapter 10
Transformation of an ASTM message	D-5
The ASTM input	D-5
XML that describes the structure of the ASTM message	D-5
Transformation with the editable XSLT file	D-8
The XML that describes the logical structure of the message	D-8
Transformation of an HL7 message	D-9
The HL7 input	D-9
XML that describes the structure of the HL7 message	D-9
Transformation with the editable XSLT file	D-13
The XML that describes the logical structure of the message	D-13

Transformation of an ASTM message

This section shows an ASTM message, as it goes through the transformations into the XML formats used internally in **cobas IT 3000**.

The ASTM input

This section shows how the following ASTM order message is transformed.

```
H|\^&|||ASTM-Host||||PSM|P||20000219111500
P|1|923608|4637463G72||Jaynes^Jane||19630101|M|||||Diabetes||||20000218102000|||||Urolog
y[CR][ETX]06[CR][LF]
O|1|923608||^NA\^CL\^K\^AMYL|||||A|||||O
L|1|F
```

Figure D-1 The raw ASTM message before any transformation

This is an order message which is read into **cobas IT 3000**.

XML that describes the structure of the ASTM message

When **cobas IT 3000** reads in the ASTM message it transforms it into an XML file that describes the structure of the ASTM message. In our example, this file is as follows. For an incoming message that **cobas IT 3000** has received, you can see the content of this file in **Administration > Communication messages > [double click log entry for incoming message] > Old content**.

```
<?xml version="1.0" encoding="utf-8"?>
<ASTM DIRECTION="IN" MODULE="LIS" SOURCE="TEST-LIS" DESTINATION="TEST-
FILE" KEY="Key not entered">
  <REC ID="H" TYPE="HEADER">
    <FIELD NO="2" TEXT="|\^&";"></FIELD>
    <FIELD NO="5" TEXT="ASTM-Host">
      <DATA>
        <COMP NO="1" TEXT="ASTM-Host"></COMP>
      </DATA>
    </FIELD>
    <FIELD NO="10" TEXT="PSM">
      <DATA>
        <COMP NO="1" TEXT="PSM"></COMP>
      </DATA>
    </FIELD>
    <FIELD NO="12" TEXT="P">
      <DATA>
        <COMP NO="1" TEXT="P"></COMP>
      </DATA>
    </FIELD>
    <FIELD NO="14" TEXT="20000219111500">
      <DATA>
        <COMP NO="1" TEXT="20000219111500"></COMP>
      </DATA>
    </FIELD>
```

Figure D-2 The XML file that describes the structure of the ASTM message

```

</REC>
<REC ID="P">
  <FIELD NO="2" TEXT="1">
    <DATA>
      <COMP NO="1" TEXT="1"></COMP>
    </DATA>
  </FIELD>
  <FIELD NO="3" TEXT="923608">
    <DATA>
      <COMP NO="1" TEXT="923608"></COMP>
    </DATA>
  </FIELD>
  <FIELD NO="4" TEXT="4637463G72">
    <DATA>
      <COMP NO="1" TEXT="4637463G72"></COMP>
    </DATA>
  </FIELD>
  <FIELD NO="6" TEXT="Jaynes^Jane">
    <DATA>
      <COMP NO="1" TEXT="Jaynes"></COMP>
      <COMP NO="2" TEXT="Jane"></COMP>
    </DATA>
  </FIELD>
  <FIELD NO="8" TEXT="19630101">
    <DATA>
      <COMP NO="1" TEXT="19630101"></COMP>
    </DATA>
  </FIELD>
  <FIELD NO="9" TEXT="M">
    <DATA>
      <COMP NO="1" TEXT="M"></COMP>
    </DATA>
  </FIELD>
  <FIELD NO="19" TEXT="Diabetes">
    <DATA>
      <COMP NO="1" TEXT="Diabetes"></COMP>
    </DATA>
  </FIELD>
  <FIELD NO="24" TEXT="20000218102000">
    <DATA>
      <COMP NO="1" TEXT="20000218102000"></COMP>
    </DATA>
  </FIELD>
  <FIELD NO="34" TEXT="Urology[CR][ETX]06[CR][LF]">
    <DATA>
      <COMP NO="1" TEXT="Urology[CR][ETX]06[CR][LF]"></COMP>
    </DATA>
  </FIELD>
  <REC ID="O">
    <FIELD NO="2" TEXT="1">
      <DATA>
        <COMP NO="1" TEXT="1"></COMP>
      </DATA>
    </FIELD>
    <FIELD NO="3" TEXT="923608">

```

Figure D-2 The XML file that describes the structure of the ASTM message

```

    <DATA>
      <COMP NO="1" TEXT="923608"></COMP>
    </DATA>
  </FIELD>
<FIELD NO="5" TEXT="^^^NA\\^^^CL\\^^^K\\^^^AMYL">
  <DATA>
    <COMP NO="4" TEXT="NA"></COMP>
  </DATA>
  <DATA>
    <COMP NO="4" TEXT="CL"></COMP>
  </DATA>
  <DATA>
    <COMP NO="4" TEXT="K"></COMP>
  </DATA>
  <DATA>
    <COMP NO="4" TEXT="AMYL"></COMP>
  </DATA>
</FIELD>
<FIELD NO="12" TEXT="A">
  <DATA>
    <COMP NO="1" TEXT="A"></COMP>
  </DATA>
</FIELD>
<FIELD NO="26" TEXT="O">
  <DATA>
    <COMP NO="1" TEXT="O"></COMP>
  </DATA>
</FIELD>
</REC>
</REC>
<REC ID="L">
  <FIELD NO="2" TEXT="1">
    <DATA>
      <COMP NO="1" TEXT="1"></COMP>
    </DATA>
  </FIELD>
  <FIELD NO="3" TEXT="F">
    <DATA>
      <COMP NO="1" TEXT="F"></COMP>
    </DATA>
  </FIELD>
</REC>
</ASTM>

```

Figure D-2 The XML file that describes the structure of the ASTM message

Transformation with the editable XSLT file

The original message has been transformed into an XML file that describes the structure of the ASTM message. It is now necessary to transform it into the XML format that **cobas IT 3000** requires, which describes the logical structure of the data. In our example, this is done with the default ASTM_IN.xsl stylesheet, as follows.

👁 For details of the default XSLT used with incoming ASTM order request messages, see *The ASTM input template* on page D-89.

The XML that describes the logical structure of the message

The final transformation creates an XML file that describes the logical structure of the data that was in the original message. For an incoming message that **cobas IT 3000** has received, you can see the content of this file in **Administration > Communication messages > [double click log entry incoming message] > Old content**. In our example, the file is as follows.

```
<?xml version="1.0" encoding="utf-8"?>
<OrderRequest>
  <MSH SENDER="HOST 3" MESSAGE_TYPE="MESSAGE_TYPE" ACCEPT_TYPE="AL" CONTROL_ID="999"></MSH>
  <PID PATID="4637463G72" LASTNAME="Jaynes" FIRSTNAME="Jane" DOB="19630101" SEX="M">
    <ORC ORDERNR="923608" SAMPLEID="923608" PRIORITY="" SPECIMEN="" ORDERER="" ORDERDATE=""
ORDERDATEFMT="" WITHDRAWDATE="" WITHDRAWDATEFMT="" STATUS="" CONTROL="NW">
      <OBR TESTCODE="NA" ACTION="A"></OBR>
      <OBR TESTCODE="CL" ACTION="A"></OBR>
      <OBR TESTCODE="K" ACTION="A"></OBR>
      <OBR TESTCODE="AMYL" ACTION="A"></OBR>
    </ORC>
  </PID>
</OrderRequest>
```

Figure D-3 The XML file that describes the logical structure of the message in the format that **cobas IT 3000** requires

The **cobas IT 3000** solution uses the XML file in this format to update its database and its internal data.

Transformation of an HL7 message

This section shows an HL7 message, as it goes through the transformations into the XML formats used internally in **cobas IT 3000**.

The HL7 input

This section shows how the following HL7order message is transformed.

```
MSH|^~\&|KIS1|||||ORM^O01|20110314191530001|||||ER|ER|
PID|||444555||Cambell^Ruth||19061213|W|
PV1|||WA1|SURG|
ORC|NW|8703560|||||R||20110314191523|WA1|||||01|
OBR||35606||102|||20110314191500||||A
OBR||35606||104|||20110314191500||||A
```

Figure D-4 The raw HL7 message before any transformation

This is an order message which is read into **cobas IT 3000**.

XML that describes the structure of the HL7 message

When **cobas IT 3000** reads in the HL7 message it transforms it into an XML file that describes the structure of the HL7 message. In our example, this file is as follows. For an incoming message that **cobas IT 3000** has received, you can see the content of this file in **Administration > Communication messages > [double click log entry incoming message] > Old content**.

```
<?xml version="1.0" encoding="UTF-8"?>
<HL7 DIRECTION="IN" MODULE="LIS" SOURCE="HIS-LIS" DESTINATION="HIS-ORDER">
  <REC ID="MSH" TYPE="HEADER">
    <FIELD NO="1" TEXT="|"/>
    <FIELD NO="2" TEXT="^~\&amp;"/>
    <FIELD NO="3" TEXT="KIS1">
      <DATA>
        <COMP NO="1" TEXT="KIS1">
          <SUB NO="1" TEXT="KIS1"/>
        </COMP>
      </DATA>
    </FIELD>
    <FIELD NO="9" TEXT="ORM^O01">
      <DATA>
        <COMP NO="1" TEXT="ORM">
          <SUB NO="1" TEXT="ORM"/>
        </COMP>
        <COMP NO="2" TEXT="O01">
          <SUB NO="1" TEXT="O01"/>
        </COMP>
      </DATA>
    </FIELD>
    <FIELD NO="10" TEXT="20110314191530001">
      <DATA>
```

Figure D-5 The XML file that describes the structure of the HL7 message

Transformation of an HL7 message

```

    <COMP NO="1" TEXT="20110314191530001">
      <SUB NO="1" TEXT="20110314191530001"/>
    </COMP>
  </DATA>
</FIELD>
<FIELD NO="15" TEXT="ER">
  <DATA>
    <COMP NO="1" TEXT="ER">
      <SUB NO="1" TEXT="ER"/>
    </COMP>
  </DATA>
</FIELD>
<FIELD NO="16" TEXT="ER">
  <DATA>
    <COMP NO="1" TEXT="ER">
      <SUB NO="1" TEXT="ER"/>
    </COMP>
  </DATA>
</FIELD>
</REC>
<REC ID="PID">
  <FIELD NO="3" TEXT="444555">
    <DATA>
      <COMP NO="1" TEXT="444555">
        <SUB NO="1" TEXT="444555"/>
      </COMP>
    </DATA>
  </FIELD>
  <FIELD NO="5" TEXT="Cambell^Ruth">
    <DATA>
      <COMP NO="1" TEXT="Cambell">
        <SUB NO="1" TEXT="Cambell"/>
      </COMP>
      <COMP NO="2" TEXT="Ruth">
        <SUB NO="1" TEXT="Ruth"/>
      </COMP>
    </DATA>
  </FIELD>
  <FIELD NO="7" TEXT="19061213">
    <DATA>
      <COMP NO="1" TEXT="19061213">
        <SUB NO="1" TEXT="19061213"/>
      </COMP>
    </DATA>
  </FIELD>
  <FIELD NO="8" TEXT="W">
    <DATA>
      <COMP NO="1" TEXT="W">
        <SUB NO="1" TEXT="W"/>
      </COMP>
    </DATA>
  </FIELD>
  <REC ID="PV1">
    <FIELD NO="3" TEXT="WA1">
      <DATA>

```

Figure D-5 The XML file that describes the structure of the HL7 message

```

        <COMP NO="1" TEXT="WA1">
            <SUB NO="1" TEXT="WA1"/>
        </COMP>
    </DATA>
</FIELD>
<FIELD NO="4" TEXT="SURG">
    <DATA>
        <COMP NO="1" TEXT="SURG">
            <SUB NO="1" TEXT="SURG"/>
        </COMP>
    </DATA>
</FIELD>
</REC>
<REC ID="ORC">
    <FIELD NO="1" TEXT="NW">
        <DATA>
            <COMP NO="1" TEXT="NW">
                <SUB NO="1" TEXT="NW"/>
            </COMP>
        </DATA>
    </FIELD>
    <FIELD NO="2" TEXT="8703560">
        <DATA>
            <COMP NO="1" TEXT="8703560">
                <SUB NO="1" TEXT="8703560"/>
            </COMP>
        </DATA>
    </FIELD>
    <FIELD NO="7" TEXT="R">
        <DATA>
            <COMP NO="1" TEXT="R">
                <SUB NO="1" TEXT="R"/>
            </COMP>
        </DATA>
    </FIELD>
    <FIELD NO="9" TEXT="20110314191523">
        <DATA>
            <COMP NO="1" TEXT="20110314191523">
                <SUB NO="1" TEXT="20110314191523"/>
            </COMP>
        </DATA>
    </FIELD>
    <FIELD NO="10" TEXT="WA1">
        <DATA>
            <COMP NO="1" TEXT="WA1">
                <SUB NO="1" TEXT="WA1"/>
            </COMP>
        </DATA>
    </FIELD>
    <FIELD NO="15" TEXT="01">
        <DATA>
            <COMP NO="1" TEXT="01">
                <SUB NO="1" TEXT="01"/>
            </COMP>
        </DATA>
    </FIELD>

```

Figure D-5 The XML file that describes the structure of the HL7 message

```

</FIELD>
<REC ID="OBR">
  <FIELD NO="2" TEXT="35606">
    <DATA>
      <COMP NO="1" TEXT="35606">
        <SUB NO="1" TEXT="35606"/>
      </COMP>
    </DATA>
  </FIELD>
  <FIELD NO="4" TEXT="102">
    <DATA>
      <COMP NO="1" TEXT="102">
        <SUB NO="1" TEXT="102"/>
      </COMP>
    </DATA>
  </FIELD>
  <FIELD NO="7" TEXT="20110314191500">
    <DATA>
      <COMP NO="1" TEXT="20110314191500">
        <SUB NO="1" TEXT="20110314191500"/>
      </COMP>
    </DATA>
  </FIELD>
  <FIELD NO="11" TEXT="A">
    <DATA>
      <COMP NO="1" TEXT="A">
        <SUB NO="1" TEXT="A"/>
      </COMP>
    </DATA>
  </FIELD>
</REC>
<REC ID="OBR">
  <FIELD NO="2" TEXT="35606">
    <DATA>
      <COMP NO="1" TEXT="35606">
        <SUB NO="1" TEXT="35606"/>
      </COMP>
    </DATA>
  </FIELD>
  <FIELD NO="4" TEXT="104">
    <DATA>
      <COMP NO="1" TEXT="104">
        <SUB NO="1" TEXT="104"/>
      </COMP>
    </DATA>
  </FIELD>
  <FIELD NO="7" TEXT="20110314191500">
    <DATA>
      <COMP NO="1" TEXT="20110314191500">
        <SUB NO="1" TEXT="20110314191500"/>
      </COMP>
    </DATA>
  </FIELD>
  <FIELD NO="11" TEXT="A">
    <DATA>

```

Figure D-5 The XML file that describes the structure of the HL7 message


```

    <COMP NO="1" TEXT="A">
      <SUB NO="1" TEXT="A"/>
    </COMP>
  </DATA>
</FIELD>
</REC>
</REC>
</REC>
</HL7>

```

Figure D-5 The XML file that describes the structure of the HL7 message

Transformation with the editable XSLT file

The original message has been transformed into an XML file that describes the structure of the HL7 message. It is now necessary to transform it into the XML format that **cobas IT 3000** requires, which describes the logical structure of the data. In our example, this is done with the default HL7_IN.xsl stylesheet.

👁 To see the default XSLT stylesheet for HL7 order requests, see *The HL7 input template* on page D-18.

The XML that describes the logical structure of the message

The final transformation creates an XML file that describes the logical structure of the data that was in the original message. For an incoming message that **cobas IT 3000** has received, you can see the content of this file in **Administration > Communication messages > [double click log entry incoming message] > Old content**. In our example, the file is as follows.

```

<OrderRequest>
  <MSH SENDER="KIS1" MESSAGE_TYPE="ORM^O01" ACCEPT_TYPE="ER" CONTROL_ID="20110314191530001"
  ACK_TYPE="ER"/>
  <PID PATID="444555" LASTNAME="Cambell" FIRSTNAME="Ruth" DOB="19061213" SEX="W" DOBFMT=""
  DATEFMT="" FLAGS="" HIS="KIS1">
    <PV1 VISITNR="" CLASS="" WARD="W1" SPECIALITY="SURG" ROOM="" ADMITDATE="" DISCHARGEDATE=""/>
    <ORC ORDERNR="8703560" SAMPLEID="35606" ORDERER="W1" ORDERDATE="20110314191523" ORDERDATEFMT=""
  WITHDRAWDATE="20110314191500" WITHDRAWDATEFMT="" HOSTDATA="200835606" PRIORITY="R" STATUS=""
  CONTROL="NW">
      <OBR TESTCODE="102" ACTION="A"/>
      <OBR TESTCODE="104" ACTION="A"/>
    </ORC>
  </PID>
</OrderRequest>

```

Figure D-6 The XML file that describes the logical structure of the message in the format that **cobas IT 3000** requires

The **cobas IT 3000** solution uses the XML file in this format to update its database and its internal data.

The default XSLT transformations

The default files used to transform messages for host communication.

This chapter gives the default XSLT files that are used to interpret and to create HL7 and ASTM messages sent to or from the host. Users can edit these files to implement their specific requirements of their site and business.

In this chapter	Chapter 11
The default XSLT templates	D-17
The HL7 templates	D-18
The HL7 input template	D-18
The HL7 output templates	D-28
The HL7 acknowledgement message response to an order request	D-28
The HL7 acknowledgement message response	D-35
The HL7 result output template	D-37
The HL7 quality control output templates	D-59
The HL7 sample event templates	D-81
The default ASTM templates	D-89
The ASTM input template	D-89
The ASTM output templates	D-96
The ASTM result report template	D-96
The ASTM quality control output template	D-105
The ASTM sample event templates	D-110

The default XSLT templates

This chapter gives you the default XSLT templates that convert:

- The XML representation of the HL7 or ASTM message to and from
- The XML representation of the data that **cobas IT 3000** uses.

You can use these stylesheets in **Workplaces > Administration > Clients > KIS > [Your HIS client] > Configuration**. You can edit them, to reconfigure how **cobas IT 3000** processes messages.

The HL7 templates

This section describes the default HL7 templates.

The HL7 input template

This section gives the default template to use with incoming order request messages from the host. This is the HL7_IN.xsl file.

You cannot send orders containing results

Although the default templates contain fields for orders containing results, this is not supported in the current release. This may be supported in future releases. If you require this functionality, please contact Roche Diagnostics Global Customer support.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!--
*****
*   DOCUMENT ELEMENT: xsl:stylesheet
*****
-->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <!-- Get rid of blank lines in output file -->
  <xsl:strip-space elements="*" />

  <xsl:output method="xml" indent="yes" version="1.0" encoding="UTF-8"/>

  <xsl:param name="acks">
    <!-- <data name="acks" type="boolean" comment="Use Ack info" /> -->
    <xsl:value-of select="true()" />
  </xsl:param>

  <xsl:param name="patid">
    <!-- <data name="patid" type="integer" comment="Position of patientid within PID segment (3
[external],4 [internal],5 [alternate])" /> -->
    <xsl:value-of select="3"/>
  </xsl:param>

  <xsl:param name="sampleid">
    <!-- <data name="sampleid" type="integer" comment="Position of sampleid within ORC segment
(3,4)" /> -->
    <xsl:value-of select="3"/>
  </xsl:param>

  <xsl:param name="ordernr">
    <!-- <data name="ordernr" type="integer" comment="Position of ordernr within ORC segment (3,4)"
/> -->
    <xsl:value-of select="2"/>
  </xsl:param>

  <xsl:param name="resultflags">
```

Figure D-7 HL7_IN.xsl

```

<!-- <data name="resultflags" type="string" comment="Result handling flags (Default: 111)" />
-->
  <xsl:value-of select="111"/>
</xsl:param>

<xsl:template match="/">
  <xsl:choose>
    <xsl:when test="/HL7/REC[@ID='MSH']/FIELD[@NO='9']/DATA/COMP/SUB/@TEXT = 'ADT'">
      <xsl:element name="PatientMessage">
        <xsl:apply-templates select="*" mode="header"/>
        <xsl:apply-templates select="*" mode="body"/>
      </xsl:element>
    </xsl:when>
    <xsl:otherwise>
      <xsl:element name="OrderRequest">
        <xsl:apply-templates select="*" mode="header"/>
        <xsl:apply-templates select="*" mode="body"/>
      </xsl:element>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>

<!-- header -->
<xsl:template match="HL7/REC[@ID='MSH']" mode="header">
  <xsl:element name="MSH">
    <xsl:attribute name="SENDER">
      <xsl:value-of select="FIELD[@NO='3']/@TEXT"/>
    </xsl:attribute>
    <xsl:if test="$acks">
      <xsl:attribute name="MESSAGE_TYPE">
        <xsl:value-of select="FIELD[@NO='9']/@TEXT"/>
      </xsl:attribute>
      <xsl:attribute name="ACCEPT_TYPE">
        <xsl:if test="FIELD[@NO='15']/@TEXT = ''">AL</xsl:if>
        <xsl:if test="FIELD[@NO='15']/@TEXT != ''">
          <xsl:value-of select="FIELD[@NO='15']/@TEXT"/>
        </xsl:if>
      </xsl:attribute>
      <xsl:attribute name="CONTROL_ID">
        <xsl:value-of select="FIELD[@NO='10']/@TEXT"/>
      </xsl:attribute>
      <xsl:attribute name="ACK_TYPE">
        <xsl:value-of select="FIELD[@NO='16']/@TEXT"/>
      </xsl:attribute>
    </xsl:if>
  </xsl:element>
</xsl:template>

<!-- body -->
<xsl:template match="REC[@ID='PID']" mode="body">
  <xsl:element name="PID">
    <xsl:attribute name="PATID">
      <xsl:value-of select="FIELD[@NO=$patid]/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="LASTNAME">

```

Figure D-7 HL7_IN.xsl

```

    <xsl:value-of select="FIELD[@NO='5']/DATA/COMP[@NO='1']/@TEXT"/>
  </xsl:attribute>
  <xsl:attribute name="FIRSTNAME">
    <xsl:value-of select="FIELD[@NO='5']/DATA/COMP[@NO='2']/@TEXT"/>
  </xsl:attribute>
  <xsl:attribute name="NAMEX">
    <xsl:value-of select="FIELD[@NO='5']/DATA/COMP[@NO='3']/@TEXT"/>
  </xsl:attribute>
  <xsl:attribute name="TITLE">
    <xsl:value-of select="FIELD[@NO='5']/DATA/COMP[@NO='5']/@TEXT"/>
  </xsl:attribute>
  <xsl:attribute name="DOB">
    <xsl:value-of select="FIELD[@NO='7']/DATA/COMP[@NO='1']/@TEXT"/>
  </xsl:attribute>
  <xsl:attribute name="SEX">
    <xsl:variable name="sex" select="FIELD[@NO='8']/DATA/COMP[@NO='1']/@TEXT"/>
    <xsl:choose>
      <xsl:when test="$sex='F'">W</xsl:when>
      <xsl:when test="$sex='W'">W</xsl:when>
      <xsl:when test="$sex='M'">M</xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="U"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:attribute>
  <xsl:attribute name="DOBFMT"/>
  <xsl:attribute name="DATEFMT"/>
  <xsl:attribute name="FLAGS"/>
  <xsl:attribute name="HIS">
    <xsl:value-of select="/HL7/REC[@ID='MSH']/FIELD[@NO='3']/DATA/COMP[@NO='1']/@TEXT"/>
  </xsl:attribute>
  <xsl:attribute name="STREET">
    <xsl:value-of select="FIELD[@NO='11']/DATA/COMP[@NO='1']/@TEXT"/>
  </xsl:attribute>
  <xsl:attribute name="CITY">
    <xsl:value-of select="FIELD[@NO='11']/DATA/COMP[@NO='3']/@TEXT"/>
  </xsl:attribute>
  <xsl:attribute name="POSTCODE">
    <xsl:value-of select="FIELD[@NO='11']/DATA/COMP[@NO='5']/@TEXT"/>
  </xsl:attribute>
  <xsl:attribute name="INSURER">
    <xsl:value-of select="FIELD[@NO='12']/DATA/COMP[@NO='1']/@TEXT"/>
  </xsl:attribute>
  <!-- MORE1="" MORE2="" MORE3="" MORE4="" MORE5="" MORE6="" MORE7="" MORE8=""
        MORE9="" MORE10="" MORE11="" MORE12="" MORE13="" MORE14="" MORE15=""
        NAMEX="" STREET="" POSTCODE="" CITY="" INSURER="" TITLE=""
        PD1="" PD2="" PD3="" PD4="" PD5="" PD6="" PD7="" PD8="" PD9="" PD10=""
        PD11="" PD12="" PD13="" PD14="" PD15="" -->
  <xsl:apply-templates select="*" mode="pv1"/>
  <xsl:apply-templates select="*" mode="orc"/>
</xsl:element>
</xsl:template>

<xsl:template match="REC[@ID='PV1']" mode="pv1">
  <xsl:element name="PV1">

```

Figure D-7 HL7_IN.xsl


```

<xsl:attribute name="VISITNR">
  <xsl:value-of select="FIELD[@NO='19']/DATA/COMP[@NO='1']/@TEXT"/>
</xsl:attribute>
<xsl:attribute name="WARD">
  <!--<xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='1']/@TEXT"/>-->
  <xsl:value-of select="FIELD[@NO='3']/@TEXT"/>
</xsl:attribute>
<xsl:attribute name="SPECIALITY">
  <xsl:value-of select="FIELD[@NO='4']/@TEXT"/>
</xsl:attribute>
<xsl:attribute name="ROOM"/>
<xsl:attribute name="ADMITDATE">
  <xsl:value-of select="FIELD[@NO='44']/DATA/COMP[@NO='1']/@TEXT"/>
</xsl:attribute>
<xsl:attribute name="DISCHARGEDATE">
  <xsl:value-of select="FIELD[@NO='45']/DATA/COMP[@NO='1']/@TEXT"/>
</xsl:attribute>
<!-- PV1="" PV2="" PV3="" PV4="" PV5="" PV6="" PV7="" PV8="" PV9="" PV10=""
      PV11="" PV12="" PV13="" PV14="" PV15=""
      <PV2 PV16="" PV17="" PV18="" PV19="" PV20=""/> -->
</xsl:element>
<xsl:apply-templates select="*" mode="dg1"/>
<xsl:apply-templates select="*" mode="in1"/>
<xsl:apply-templates select="*" mode="orc"/>
</xsl:template>

<xsl:template match="REC[@ID='DG1']" mode="dg1">
  <xsl:element name="DG1">
    <xsl:attribute name="ICDCODE">
      <xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='2']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="TYPE">
      <xsl:value-of select="FIELD[@NO='6']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="DATE">
      <xsl:value-of select="FIELD[@NO='5']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="DATEFMT">
      <!-- YYYYMMDDHHMM-->
    </xsl:attribute>
  </xsl:element>
  <!-- <DG1 DG1="" DG2="" DG3="" DG4="" DG5="" DG6="" DG7="" DG8="" DG9="" DG10=""
      DG11="" DG12="" DG13="" DG14="" DG15="" DG16="" DG17="" DG18="" DG19="" DG20=""/> -->
</xsl:template>

<xsl:template match="REC[@ID='IN1']" mode="in1">
  <xsl:element name="IN1">
    <xsl:attribute name="IN1">
      <xsl:value-of select="FIELD[@NO='3']/@TEXT"/>
    </xsl:attribute>
    <xsl:element name="IN2">
    </xsl:element>
  </xsl:element>

```

Figure D-7 HL7_IN.xsl

```

</xsl:element>
<!--<IN1 IN1="" IN2="" IN3="" IN4="" IN5="" IN6="" IN7="" IN8="" IN9="" IN10="" IN11="" IN12=""
IN13="" IN14="" IN15="">
    <IN2 IN16="" IN17="" IN18="" IN19="" IN20=""/> -->
</xsl:template>

<xsl:template match="REC[@ID='ORC']" mode="orc">
    <xsl:element name="ORC">
        <xsl:attribute name="ORDERNR">
            <!-- substring parameter should start with 1, our implementation starts with 0-->
            <!-- just take the last to digits as specimen code and split it up -->
            <!--
                <xsl:value-of select="substring(REC[@ID='OBR']/
FIELD[@NO=$ordernr]/DATA/COMP[@NO='1']/@TEXT, 1, string-length(REC[@ID='OBR']/FIELD[@NO=$ordernr]/
DATA/COMP[@NO='1']/@TEXT))"/>
            -->

            <!--this is added to receive the order number as per msg sent 28.09.2009 Boon -->
            <xsl:value-of select="FIELD[@NO=$ordernr]/DATA/COMP[@NO='1']/@TEXT"/>
        </xsl:attribute>
        <xsl:attribute name="SAMPLEID">
            <!--why is this being hardcoded in stylesheet? should commented this line-->
            <!--
                <xsl:value-of select="concat(substring(REC[@ID='OBR']/FIELD[@NO=$ordernr]/DATA/COMP[@NO='1']/
@TEXT, 1, string-length(REC[@ID='OBR']/FIELD[@NO=$ordernr]/DATA/COMP[@NO='1']/@TEXT)), '01')"/>
            -->
            <!-- this is required to read the sample ID field based on the selection in the
configuration 28.09.09 -->
            <!--
                <xsl:value-of select="FIELD[@NO='6']/DATA/COMP[@NO='1']/@TEXT"/>
            -->

            <xsl:value-of select="REC[@ID='OBR']/FIELD[@NO=$sampleid]/DATA/COMP[@NO='1']/@TEXT"/>
        </xsl:attribute>
        <xsl:attribute name="LOCATION">
            <xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='1']/@TEXT"/>
        </xsl:attribute>
        <xsl:attribute name="DESTINATION">
            <xsl:value-of select="FIELD[@NO='4']/DATA/COMP[@NO='1']/@TEXT"/>
        </xsl:attribute>
        <!--
        <xsl:attribute name="SPECIMEN">
            <xsl:value-of select="FIELD[@NO='8']/DATA/COMP[@NO='1']/@TEXT"/>
        </xsl:attribute>
    -->

    <xsl:attribute name="PHYSICIAN_CREATE">
        <xsl:value-of select="FIELD[@NO='13']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="PHYSICIAN_CODE">
        <xsl:value-of select="FIELD[@NO='12']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="PHYSICIAN_NAME">
        <xsl:value-of select="FIELD[@NO='11']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <!-- ORDERER, is it hte 10th field? -->
    <xsl:attribute name="ORDERER">
        <xsl:value-of select="FIELD[@NO='10']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>

```

Figure D-7 HL7_IN.xsl

```

<xsl:attribute name="ORDERDATE">
  <!-- this was previously commented out, we need this field to know the orderdate-->
  <xsl:value-of select="FIELD[@NO='9']/DATA/COMP[@NO='1']/@TEXT"/>
</xsl:attribute>
<xsl:attribute name="ORDERDATEFMT">
  <xsl:value-of select="FIELD[@NO='9']/DATA/COMP[@NO='2']/@TEXT"/>
</xsl:attribute>
<xsl:attribute name="WITHDRAWDATE">
  <xsl:value-of select="REC[@ID='OBR']/FIELD[@NO=7]/DATA/COMP[@NO='1']/@TEXT"/>
</xsl:attribute>
<xsl:attribute name="WITHDRAWDATEFMT">
  <xsl:value-of select="REC[@ID='OBR']/FIELD[@NO=7]/DATA/COMP[@NO='2']/@TEXT"/>
</xsl:attribute>
<xsl:attribute name="HOSTDATA">
  <xsl:value-of select="concat('2008', substring(REC[@ID='OBR']/FIELD[@NO=$ordernr]/DATA/
COMP[@NO='1']/@TEXT, 1, string-length(REC[@ID='OBR']/FIELD[@NO=$ordernr]/DATA/COMP[@NO='1']/
@TEXT)))"/>
</xsl:attribute>
<xsl:attribute name="PRIORITY">
  <!--before 16.11.2009 Boon: Existing code which accept whatever was transmitted by the
host
  <xsl:value-of select="FIELD[@NO='7']/DATA/COMP[@NO='1']/@TEXT"/>
  -->
  <!-- 16.11.2009 Boon: Priority has to be set to S & R in cobas IT Application. And if
there is no priority sent, it will be defaulted to R-Routine.-->
  <xsl:variable name="priority" select="FIELD[@NO='7']/DATA/COMP[@NO='1']/@TEXT"/>
  <xsl:choose>
    <xsl:when test="$priority='S'">S</xsl:when>
    <xsl:when test="$priority='R'">R</xsl:when>
    <xsl:otherwise>R</xsl:otherwise>
  </xsl:choose>
</xsl:attribute>

  <!--
  <xsl:attribute name="SPECIMEN">
    <xsl:value-of select="FIELD[@NO='15']/DATA/COMP[@NO='1']/@TEXT"/>
  </xsl:attribute>
  -->

  <xsl:attribute name="STATUS">
    <xsl:variable name="status" select="FIELD[@NO='5']/DATA/COMP[@NO='1']/@TEXT"/>
    <xsl:choose>
      <!-- Some, but not all, results are available -->
      <xsl:when test="$status='A'">A</xsl:when>
      <!-- Order was cancelled -->
      <xsl:when test="$status='CA'">CA</xsl:when>
      <!-- In process, scheduled -->
      <xsl:when test="$status='SC'">SC</xsl:when>
      <!-- Order has been replaced -->
      <xsl:when test="$status='R'">RP</xsl:when>
      <xsl:when test="$status='RP'">RP</xsl:when>
      <xsl:when test="$status='UP'">UP</xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="$status"/>
      </xsl:otherwise>
    <!-- Default empty -->

```

Figure D-7 HL7_IN.xsl

```

</xsl:choose>
</xsl:attribute>
<xsl:attribute name="CONTROL">
  <xsl:variable name="control" select="FIELD[@NO='1']/DATA/COMP[@NO='1']/@TEXT"/>
  <xsl:choose>
    <xsl:when test="$control='NW'">NW</xsl:when>
    <!-- New Order -->
    <xsl:when test="$control='XO'">XO</xsl:when>
    <!-- Changed Order -->
    <xsl:when test="$control='CA'">CA</xsl:when>
    <!-- Canceled Order
    06-09-08/alo: chandeg from NW

    -->
    <xsl:otherwise>
      <xsl:value-of select="$control"/>
    </xsl:otherwise>
    <!-- Default New Order -->
  </xsl:choose>
</xsl:attribute>

<xsl:apply-templates select="*" mode="NTE"/>

<xsl:for-each select="REC[@ID='OBR']">
  <xsl:element name="OBR">

    <xsl:variable name="Testcode" select="FIELD[@NO='4']/DATA/COMP/@TEXT"/>

    <xsl:attribute name="TESTCODE">
      <xsl:value-of select="$Testcode"/>
    </xsl:attribute>

    <xsl:attribute name="SAMPLEID">
      <xsl:value-of select="FIELD[@NO='3']/DATA/COMP/@TEXT"/>
    </xsl:attribute>

    <xsl:attribute name="SPECIMEN">
      <xsl:value-of select="FIELD[@NO='5']/DATA/COMP/@TEXT"/>
    </xsl:attribute>

    <xsl:attribute name="ACTION">
      <xsl:variable name="action" select="FIELD[@NO='11']/DATA/COMP[@NO='1']/@TEXT"/>
      <xsl:choose>
        <!-- specimen action code -->
        <!-- add ordered tests to existing specimen -->
        <xsl:when test="$action='A'">A</xsl:when>
        <!-- generated order, reflex order -->
        <xsl:when test="$action='G'">G</xsl:when>
        <!-- revised order -->
        <xsl:when test="$action='R'">R</xsl:when>
        <!-- THESE codes are not valid in HL7 !!!
        <xsl:when test="$action='C'">C</xsl:when>
        <xsl:when test="$action='N'">C</xsl:when> -->
        <!-- Cancel -->
        <!-- default: add to specimen -->

```

Figure D-7 HL7_IN.xsl

```

        <xsl:otherwise>
            <xsl:value-of select="$action"/>
        </xsl:otherwise>
    </xsl:choose>
</xsl:attribute>

    <xsl:apply-templates select="./REC[@ID='OBX']" mode="OBX">
        <xsl:with-param name="Testcode" select="$Testcode"/>
    </xsl:apply-templates>

</xsl:element>
</xsl:for-each>
</xsl:element>
</xsl:template>

<!--
*****
*   Add OBX to OBR
*****
-->

<xsl:template match="*" mode="OBX">
    <xsl:param name="Testcode"/>

    <xsl:variable name="TestcodeOBX">
        <xsl:value-of select="./FIELD[@NO='3']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:variable>

    <!-- The HL7 standard allows to map every incoming OBX to the parent    -->
    <!-- OBR regardless of the testcode. Therefore we can map without        -->
    <!-- comparing the testcode. This allows the receival of result also for -->
    <!-- profiles. This stylesheet will generate the following structure:    -->
    <!-- <OBR TESTCODE="YourProfileCode" ...>                                -->
    <!--     <OBX Testcode="AnalyteCode".../>                                -->
    <!-- </OBR>                                                                -->
    <!-- The backend will handle all OBX childs of an OBR and will use the  -->
    <!-- testcode of the OBX to insert the result                          -->
    <!-- If you want to prevent this behavior and compare the testcode of the -->
    <!-- OBR to the one in the OBX comment in the <xsl:if> statement        -->

    <!--xsl:if test="$TestcodeOBX = $Testcode" -->
        <xsl:element name="OBX">

            <xsl:attribute name="TESTCODE">
                <xsl:value-of select="$TestcodeOBX"/>
            </xsl:attribute>

            <xsl:attribute name="RESULT">
                <xsl:value-of select="FIELD[@NO='5']/DATA/COMP[@NO='1']/@TEXT"/>
            </xsl:attribute>

            <xsl:attribute name="UNIT">
                <xsl:value-of select="FIELD[@NO='6']/DATA/COMP[@NO='1']/@TEXT"/>
            </xsl:attribute>

            <xsl:attribute name="REFRANGE">

```

Figure D-7 HL7_IN.xsl

```

        <xsl:value-of select="FIELD[@NO='7']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>

    <xsl:attribute name="INDICATOR">
        <xsl:value-of select="FIELD[@NO='8']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>

    <xsl:attribute name="RESULTFLAGS">
        <xsl:value-of select="$resultflags"/>
    </xsl:attribute>

    <xsl:attribute name="STATUS">
        <xsl:choose>
            <!-- R: result is already validated and is NOT validated internally, -->
            <!-- validation status 'Single Order' is set -->
            <!-- X: deprecated, LIS_APP offers other possibilities to delete a -->
            <!-- test -->
            <!-- anything else: result is internally validated. It is recommended -->
            <!-- to use the test status '3', other values may have side effects -->
            <xsl:when test="1=1">R</xsl:when>
            <xsl:otherwise>R</xsl:otherwise>
        </xsl:choose>
    </xsl:attribute>
</xsl:element>
<!--/xsl:if-->
</xsl:template>

<!--
*****
* Add sample comments C1 .. C5
*****
-->

<xsl:template match="REC[@ID='NTE']" mode="NTE">
    <xsl:if test="FIELD[@NO='3']/DATA/COMP[@NO='1']/@TEXT">
        <xsl:element name="NTE">
            <xsl:attribute name="TEXT">
                <xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='1']/@TEXT"/>
            </xsl:attribute>
        </xsl:element>
    </xsl:if>
    <xsl:if test="FIELD[@NO='3']/DATA/COMP[@NO='2']/@TEXT">
        <xsl:element name="NTE">
            <xsl:attribute name="TEXT">
                <xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='2']/@TEXT"/>
            </xsl:attribute>
        </xsl:element>
    </xsl:if>
    <xsl:if test="FIELD[@NO='3']/DATA/COMP[@NO='3']/@TEXT">
        <xsl:element name="NTE">
            <xsl:attribute name="TEXT">
                <xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='3']/@TEXT"/>
            </xsl:attribute>
        </xsl:element>
    </xsl:if>
    <xsl:if test="FIELD[@NO='3']/DATA/COMP[@NO='4']/@TEXT">

```

Figure D-7 HL7_IN.xsl

```

        <xsl:element name="NTE">
            <xsl:attribute name="TEXT">
                <xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='4']/@TEXT"/>
            </xsl:attribute>
        </xsl:element>
    </xsl:if>
    <xsl:if test="FIELD[@NO='3']/DATA/COMP[@NO='5']/@TEXT">
        <xsl:element name="NTE">
            <xsl:attribute name="TEXT">
                <xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='5']/@TEXT"/>
            </xsl:attribute>
        </xsl:element>
    </xsl:if>
    <xsl:if test="FIELD[@NO='3']/DATA/COMP[@NO='6']/@TEXT">
        <xsl:element name="NTE">
            <xsl:attribute name="TEXT">
                <xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='6']/@TEXT"/>
            </xsl:attribute>
        </xsl:element>
    </xsl:if>
    <xsl:if test="FIELD[@NO='3']/DATA/COMP[@NO='7']/@TEXT">
        <xsl:element name="NTE">
            <xsl:attribute name="TEXT">
                <xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='7']/@TEXT"/>
            </xsl:attribute>
        </xsl:element>
    </xsl:if>
    <xsl:if test="FIELD[@NO='3']/DATA/COMP[@NO='8']/@TEXT">
        <xsl:element name="NTE">
            <xsl:attribute name="TEXT">
                <xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='8']/@TEXT"/>
            </xsl:attribute>
        </xsl:element>
    </xsl:if>
</xsl:template>
<!--
*****
*   END OF DOCUMENT
*****
-->
</xsl:stylesheet>

```

Figure D-7 HL7_IN.xsl

The HL7 output templates

This section describes the default XSLT templates for HL7 outgoing messages sent to the host.

The HL7 acknowledgement message response to an order request

This section gives the XSLT file used to create the HL7 acknowledgement sent in response to an order request. This file is HL7_ACK_OrderResponse.xsl

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!--
*****
*DOCUMENT ELEMENT: xsl:stylesheet
*****
-->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" indent="yes" version="1.0" encoding="UTF-8"/>
  <xsl:param name="SENDING_APPLICATION">
    <!-- <data name="SENDING_APPLICATION" comment="The name of the sending application" type="string"
  /> -->
    <xsl:value-of select="'"'/>
  </xsl:param>
  <xsl:param name="SENDING_FACILITY">
    <!-- <data name="SENDING_FACILITY" comment="The name of the sending facility" type="string" /> -
  -->
    <xsl:value-of select="'"'/>
  </xsl:param>
  <xsl:param name="RECEIVING_APPLICATION">
    <!-- <data name="RECEIVING_APPLICATION" comment="The name of the receiving application"
type="string" /> -->
    <xsl:value-of select="'"'/>
  </xsl:param>
  <xsl:param name="RECEIVING_FACILITY">
    <!-- <data name="RECEIVING_FACILITY" comment="The name of the receiving facility" type="string"
  /> -->
    <xsl:value-of select="'"'/>
  </xsl:param>

  <!-- Local variables -->
  <xsl:param name="L_SENDING_APPLICATION">
    <xsl:choose>
      <xsl:when test="string-length($SENDING_APPLICATION) != 0">
        <xsl:value-of select="$SENDING_APPLICATION"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="//MSH/@SENDING_APPLICATION"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:param>
  <xsl:param name="L_SENDING_FACILITY">
    <xsl:choose>
      <xsl:when test="string-length($SENDING_FACILITY) != 0">
        <xsl:value-of select="$SENDING_FACILITY"/>
      </xsl:when>
```

Figure D-8 HL7_ACK_OrderResponse.xsl


```

    <xsl:otherwise>
      <xsl:value-of select="//MSH/@SENDING_FACILITY"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:param>
<xsl:param name="L_RECEIVING_APPLICATION">
  <xsl:choose>
    <xsl:when test="string-length($RECEIVING_APPLICATION) != 0">
      <xsl:value-of select="$RECEIVING_APPLICATION"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="//MSH/@RECEIVING_APPLICATION"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:param>
<xsl:param name="L_RECEIVING_FACILITY">
  <xsl:choose>
    <xsl:when test="string-length($RECEIVING_FACILITY) != 0">
      <xsl:value-of select="$RECEIVING_FACILITY"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="//MSH/@RECEIVING_FACILITY"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:param>
<xsl:template match="/">
  <xsl:element name="HL7">
    <xsl:attribute name="DIRECTION">OUT</xsl:attribute>
    <xsl:attribute name="MODULE">cobasIT3000</xsl:attribute>
    <xsl:apply-templates select="*" mode="header"/>
    <xsl:apply-templates select="*" mode="body"/>
    <xsl:apply-templates select="*" mode="footer"/>
  </xsl:element>
</xsl:template>

<!-- header -->
<xsl:template match="MSH" mode="header">
  <xsl:element name="REC">
    <xsl:attribute name="ID">MSH</xsl:attribute>
    <xsl:attribute name="TYPE">HEADER</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">|</xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">^&amp;~\</xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_SENDING_APPLICATION"/>
      </xsl:attribute>
      <xsl:element name="DATA">
        <xsl:element name="COMP">

```

Figure D-8 HL7_ACK_OrderResponse.xsl

```

    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="$L_SENDING_APPLICATION"/>
    </xsl:attribute>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_SENDING_APPLICATION"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">4</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="$L_SENDING_FACILITY"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_SENDING_FACILITY"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_SENDING_FACILITY"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">5</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="$L_RECEIVING_APPLICATION"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_RECEIVING_APPLICATION"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_RECEIVING_APPLICATION"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">6</xsl:attribute>

```

Figure D-8 HL7_ACK_OrderResponse.xsl

```

<xsl:attribute name="TEXT">
  <xsl:value-of select="$L_RECEIVING_FACILITY"/>
</xsl:attribute>
<xsl:element name="DATA">
  <xsl:element name="COMP">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="$L_RECEIVING_FACILITY"/>
    </xsl:attribute>
  </xsl:element>
  <xsl:element name="SUB">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="$L_RECEIVING_FACILITY"/>
    </xsl:attribute>
  </xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">7</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@DATE"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@DATE"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@DATE"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">9</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@MESSAGE_TYPE"/>^<xsl:value-of select="@EVENT_TYPE"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@MESSAGE_TYPE"/>
      </xsl:attribute>
    <!-- <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@MESSAGE_TYPE"/>
      </xsl:attribute>
    </xsl:element> -->
  </xsl:element>

```

Figure D-8 HL7_ACK_OrderResponse.xsl

```

    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@EVENT_TYPE"/>
      </xsl:attribute>
      <!--<xsl:element name="SUB">
        <xsl:attribute name="NO">2</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@EVENT_TYPE"/>
        </xsl:attribute>
      </xsl:element> -->
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">10</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@CONTROL_ID"/>
  </xsl:attribute>
  <xsl:element name="COMP">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@CONTROL_ID"/>
    </xsl:attribute>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@CONTROL_ID"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">11</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@PROCESSING_ID"/>
  </xsl:attribute>
  <xsl:element name="COMP">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@PROCESSING_ID"/>
    </xsl:attribute>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@PROCESSING_ID"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">12</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@VERSION"/>
  </xsl:attribute>

```

Figure D-8 HL7_ACK_OrderResponse.xsl

```

</xsl:attribute>
<xsl:element name="COMP">
  <xsl:attribute name="NO">1</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@VERSION"/>
  </xsl:attribute>
  <xsl:element name="SUB">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@VERSION"/>
    </xsl:attribute>
  </xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">15</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@ACCEPT_TYPE"/>
  </xsl:attribute>
  <xsl:element name="COMP">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@ACCEPT_TYPE"/>
    </xsl:attribute>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ACCEPT_TYPE"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">16</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@ACK_TYPE"/>
  </xsl:attribute>
  <xsl:element name="COMP">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@ACK_TYPE"/>
    </xsl:attribute>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ACK_TYPE"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">18</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@ENCODING"/>
  </xsl:attribute>

```

Figure D-8 HL7_ACK_OrderResponse.xsl

```

    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ENCODING"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@ENCODING"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:template>
<!-- body -->
<xsl:template match="MSA" mode="body">
  <xsl:element name="REC">
    <xsl:attribute name="ID">MSA</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ACK"/>
      </xsl:attribute>
      <xsl:element name="DATA">
        <xsl:element name="COMP">
          <xsl:attribute name="NO">1</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@ACK"/>
          </xsl:attribute>
        </xsl:element>
      </xsl:element>
    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">2</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@SEQ"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@SEQ"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">3</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@TEXT"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">

```

Figure D-8 HL7_ACK_OrderResponse.xsl

```

        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
            <xsl:value-of select="@TEXT"/>
        </xsl:attribute>
    </xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">4</xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">5</xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">6</xsl:attribute>
</xsl:element>
</xsl:element>
</xsl:template>

<!-- footer -->
<xsl:template match="MSH" mode="footer">
    <xsl:apply-templates select="*" mode="footer"/>
</xsl:template>
<xsl:template match="*" mode="header">
    <xsl:apply-templates select="*" mode="header"/>
</xsl:template>
<xsl:template match="*" mode="body">
    <xsl:apply-templates select="*" mode="body"/>
</xsl:template>
<xsl:template match="*" mode="footer">
    <xsl:apply-templates select="*" mode="footer"/>
</xsl:template>
<!--*****
*END OF DOCUMENT
*****-->
</xsl:stylesheet>

```

Figure D-8 HL7_ACK_OrderResponse.xsl

The HL7 acknowledgement message response

This section gives the XSLT file used to create the HL7 acknowledgement sent to all messages except an order request. This file is HL7_ACK_MessageResponse.xsl.

```

<?xml version='1.0' encoding='utf-8'?>
<!--
*****
* DOCUMENT ELEMENT: xsl:stylesheet
*****
-->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output method="html" indent="yes" version="1.0" encoding="UTF-8"/>
    <xsl:template match="/">
        <xsl:element name="ResponseMessage">

```

Figure D-9 HL7_ACK_MessageResponse.xsl

The HL7 templates

```

<!-- <xsl:apply-templates select="*" mode="header"/> -->
<xsl:apply-templates select="*" />

<!-- <xsl:apply-templates select="*" mode="body"/> -->

</xsl:element>

</xsl:template>

<!-- header -->
<!-- <xsl:template match="HL7/REC[@ID='MSH']" mode="header"> -->
<xsl:template match="HL7/REC[@ID='MSH']">
  <xsl:element name="MSH">
    <xsl:attribute name="SENDER">
      <xsl:value-of select="FIELD[@NO='3']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="MESSAGE_TYPE">
      <xsl:value-of select="FIELD[@NO='9']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="ACCEPT_TYPE">
      <xsl:value-of select="FIELD[@NO='15']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="CONTROL_ID">
      <xsl:value-of select="FIELD[@NO='10']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="ACK_TYPE">
      <xsl:value-of select="FIELD[@NO='16']/@TEXT"/>
    </xsl:attribute>
  </xsl:element>
<!--<xsl:apply-templates select="*" mode="body"/>-->
</xsl:template>
<!-- body -->
<!--<xsl:template match="REC[@ID='MSA']" mode="body"> -->
<xsl:template match="REC[@ID='MSA']">
  <xsl:element name="MSA">
    <xsl:attribute name="SEQ">

      <xsl:value-of select="FIELD[@NO='2']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="ACK">
      <xsl:value-of select="FIELD[@NO='1']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="FIELD[@NO='3']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
  </xsl:element>
</xsl:template>
<!--
*****
*   END OF DOCUMENT
*****
-->
</xsl:stylesheet>

```

Figure D-9 HL7_ACK_MessageResponse.xsl

The HL7 result output template

This section gives the XSLT that **cobas IT 3000** uses to create result report messages sent to the host. This file is HL7_OUT.xml.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!--
*****
*DOCUMENT ELEMENT: xsl:stylesheet
*****
-->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:param name="patid_pos">
    <!-- <data name="patid" type="integer" comment="Position of patientid within PID segment (3
[external],4 [internal],5 [alternate])"/> -->
    <xsl:value-of select="3"/>
  </xsl:param>
  <xsl:param name="posSID">
    <!-- <data name="posSID" type="boolean" comment="True :to have Sample ID, otherwise Order NR at
OBR section" /> -->
    <xsl:value-of select="true()"/>
  </xsl:param>
  <xsl:param name="SENDING_APPLICATION">
    <!-- <data name="SENDING_APPLICATION" comment="The name of the sending application"
type="string" /> -->
    <xsl:value-of select=""/>
  </xsl:param>
  <xsl:param name="SENDING_FACILITY">
    <!-- <data name="SENDING_FACILITY" comment="The name of the sending facility" type="string" /> -
-->
    <xsl:value-of select=""/>
  </xsl:param>
  <xsl:param name="RECEIVING_APPLICATION">
    <!-- <data name="RECEIVING_APPLICATION" comment="The name of the receiving application"
type="string" /> -->
    <xsl:value-of select=""/>
  </xsl:param>
  <xsl:param name="RECEIVING_FACILITY">
    <!-- <data name="RECEIVING_FACILITY" comment="The name of the receiving facility" type="string"
/> -->
    <xsl:value-of select=""/>
  </xsl:param>
  <xsl:param name="ACCEPT_TYPE">
    <!-- <data name="ACCEPT_TYPE" comment="[MSH 15] The type of acknowledge , AL (Always), NE (Never)
,ER (Error) , SU (Successful)" type="string" /> -->
    <xsl:value-of select=""/>
  </xsl:param>
  <xsl:param name="ACK_TYPE">
    <!-- <data name="ACK_TYPE" comment="[MSH 16] The type of application acknowledge , AL (Always),
NE (Never) ,ER (Error) , SU (Successful)" type="string" /> -->
    <xsl:value-of select=""/>
  </xsl:param>
```

Figure D-10 HL7_OUT.xml

```

<!-- Local variables -->
<xsl:param name="L_SENDING_APPLICATION">
  <xsl:choose>
    <xsl:when test="string-length($SENDING_APPLICATION) != 0">
      <xsl:value-of select="$SENDING_APPLICATION"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="//MSH/@SENDING_APPLICATION"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:param>
<xsl:param name="L_SENDING_FACILITY">
  <xsl:choose>
    <xsl:when test="string-length($SENDING_FACILITY) != 0">
      <xsl:value-of select="$SENDING_FACILITY"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="//MSH/@SENDING_FACILITY"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:param>
<xsl:param name="L_RECEIVING_APPLICATION">
  <xsl:choose>
    <xsl:when test="string-length($RECEIVING_APPLICATION) != 0">
      <xsl:value-of select="$RECEIVING_APPLICATION"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="//MSH/@RECEIVING_APPLICATION"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:param>
<xsl:param name="L_RECEIVING_FACILITY">
  <xsl:choose>
    <xsl:when test="string-length($RECEIVING_FACILITY) != 0">
      <xsl:value-of select="$RECEIVING_FACILITY"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="//MSH/@RECEIVING_FACILITY"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:param>
<xsl:param name="L_ACCEPT_TYPE">
  <xsl:choose>
    <xsl:when test="string-length($ACCEPT_TYPE) != 0">
      <xsl:value-of select="$ACCEPT_TYPE"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="//MSH/@ACCEPT_TYPE"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:param>
<xsl:param name="L_ACK_TYPE">
  <xsl:choose>
    <xsl:when test="string-length($ACK_TYPE) != 0">

```

Figure D-10 HL7_OUT.xsl

```

        <xsl:value-of select="$ACK_TYPE"/>
    </xsl:when>
    <xsl:otherwise>
        <xsl:value-of select="//MSH/@ACK_TYPE"/>
    </xsl:otherwise>
    </xsl:choose>
</xsl:param>

<xsl:template match="/">
    <xsl:element name="HL7">
        <xsl:attribute name="DIRECTION">OUT</xsl:attribute>
        <xsl:attribute name="MODULE">cobasIT3000</xsl:attribute>
        <xsl:apply-templates select="*" mode="header"/>
        <xsl:apply-templates select="*" mode="body"/>
        <xsl:apply-templates select="*" mode="footer"/>
    </xsl:element>
</xsl:template>
<!-- header -->
<!--
*****
*           MSH Template (header)                               *
*****
-->
<xsl:template match="MSH" mode="header">
    <xsl:element name="REC">
        <xsl:attribute name="ID">MSH</xsl:attribute>
        <xsl:attribute name="TYPE">HEADER</xsl:attribute>
        <xsl:element name="FIELD">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">|</xsl:attribute>
        </xsl:element>
        <xsl:element name="FIELD">
            <xsl:attribute name="NO">2</xsl:attribute>
            <xsl:attribute name="TEXT">^~\&amp;</xsl:attribute>
        </xsl:element>
        <xsl:element name="FIELD">
            <xsl:attribute name="NO">3</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="$L_SENDING_APPLICATION"/>
            </xsl:attribute>
            <xsl:element name="DATA">
                <xsl:element name="COMP">
                    <xsl:attribute name="NO">1</xsl:attribute>
                    <xsl:attribute name="TEXT">
                        <xsl:value-of select="$L_SENDING_APPLICATION"/>
                    </xsl:attribute>
                    <xsl:element name="SUB">
                        <xsl:attribute name="NO">1</xsl:attribute>
                        <xsl:attribute name="TEXT">
                            <xsl:value-of select="$L_SENDING_APPLICATION"/>
                        </xsl:attribute>
                    </xsl:element>
                </xsl:element>
            </xsl:element>
        </xsl:element>
    </xsl:element>

```

Figure D-10 HL7_OUT.xsl

```

<xsl:element name="FIELD">
  <xsl:attribute name="NO">4</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="$L_SENDING_FACILITY"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_SENDING_FACILITY"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="$L_SENDING_FACILITY"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">5</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="$L_RECEIVING_APPLICATION"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_RECEIVING_APPLICATION"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="$L_RECEIVING_APPLICATION"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">6</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="$L_RECEIVING_FACILITY"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_RECEIVING_FACILITY"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="$L_RECEIVING_FACILITY"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:element>

```

Figure D-10 HL7_OUT.xsl

```

        </xsl:attribute>
    </xsl:element>
</xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">7</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@DATE"/>
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@DATE"/>
            </xsl:attribute>
            <xsl:element name="SUB">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">
                    <xsl:value-of select="@DATE"/>
                </xsl:attribute>
            </xsl:element>
        </xsl:element>
    </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">9</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@MESSAGE_TYPE" />
        ^
        <xsl:value-of select="@EVENT_TYPE" />
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@MESSAGE_TYPE"/>
            </xsl:attribute>
        <!--
    <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@MESSAGE_TYPE" />
    </xsl:attribute>
    </xsl:element>
-->
    </xsl:element>
    <xsl:element name="COMP">
        <xsl:attribute name="NO">2</xsl:attribute>
        <xsl:attribute name="TEXT">
            <xsl:value-of select="@EVENT_TYPE"/>
        </xsl:attribute>
        <!-- <xsl:element name="SUB">
    <xsl:attribute name="NO">2</xsl:attribute>
    <xsl:attribute name="TEXT">

```

Figure D-10 HL7_OUT.xsl

```

<xsl:value-of select="@EVENT_TYPE" />
</xsl:attribute>
</xsl:element>  -->
    </xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">10</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@CONTROL_ID"/>
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@CONTROL_ID"/>
            </xsl:attribute>
            <xsl:element name="SUB">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">
                    <xsl:value-of select="@CONTROL_ID"/>
                </xsl:attribute>
            </xsl:element>
        </xsl:element>
    </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">11</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@PROCESSING_ID"/>
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@PROCESSING_ID"/>
            </xsl:attribute>
            <xsl:element name="SUB">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">
                    <xsl:value-of select="@PROCESSING_ID"/>
                </xsl:attribute>
            </xsl:element>
        </xsl:element>
    </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">12</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@VERSION"/>
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">

```

Figure D-10 HL7_OUT.xsl

```

        <xsl:value-of select="@VERSION"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@VERSION"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">15</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="$L_ACCEPT_TYPE"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_ACCEPT_TYPE"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="$L_ACCEPT_TYPE"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">16</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="$L_ACK_TYPE"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_ACK_TYPE"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="$L_ACK_TYPE"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">18</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@ENCODING"/>
  </xsl:attribute>

```

Figure D-10 HL7_OUT.xsl

```

</xsl:attribute>
<xsl:element name="DATA">
  <xsl:element name="COMP">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@ENCODING"/>
    </xsl:attribute>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ENCODING"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element>
  <xsl:apply-templates select="*" mode="header"/>
</xsl:element>
</xsl:template>
<!--
*****
*           Body Template                               *
*****
-->
<xsl:template match="PID" mode="body">
  <xsl:element name="REC">
    <xsl:attribute name="ID">PID</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SET"/>
      </xsl:attribute>
      <xsl:element name="DATA">
        <xsl:element name="COMP">
          <xsl:attribute name="NO">1</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@SET"/>
          </xsl:attribute>
        </xsl:element>
      </xsl:element>
    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">
      <xsl:value-of select="$patid_pos"/>
    </xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@PATID"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@PATID"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>

```

Figure D-10 HL7_OUT.xsl


```

    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">5</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@LASTNAME" />
      ^
      <xsl:value-of select="@FIRSTNAME" />
    </xsl:attribute>
  </xsl:element>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@LASTNAME"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@FIRSTNAME"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">7</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@DOB"/>
  </xsl:attribute>
</xsl:element>
<xsl:element name="DATA">
  <xsl:element name="COMP">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@DOB"/>
    </xsl:attribute>
  </xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">8</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@SEX"/>
  </xsl:attribute>
</xsl:element>
<xsl:element name="DATA">
  <xsl:element name="COMP">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@SEX"/>
    </xsl:attribute>
  </xsl:element>
</xsl:element>
</xsl:element>
<xsl:apply-templates select="*" mode="pv1"/>
<xsl:apply-templates select="*" mode="orc"/>
</xsl:element>

```

Figure D-10 HL7_OUT.xsl

```

</xsl:template>
<!--
*****
*          PV1 Template                                     *
*****
-->
<xsl:template match="PV1" mode="pv1">
  <xsl:element name="REC">
    <xsl:attribute name="ID">PV1</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SET"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@SET"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">19</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@VISITNR"/>
    </xsl:attribute>
  </xsl:element>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@VISITNR"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">44</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@ADMITDATE"/>
    </xsl:attribute>
  </xsl:element>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ADMITDATE"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">45</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@DISCHARGEDATE"/>
    </xsl:attribute>
  </xsl:element>

```

Figure D-10 HL7_OUT.xsl

```

    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@DISCHARGEDATE"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:template>

<xsl:template match="ORC" mode="orc">
  <xsl:element name="REC">
    <xsl:attribute name="ID">ORC</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <!-- 1-->
        <xsl:value-of select="@SET" />
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ORDERNR"/>
      </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@ORDERNR"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">9</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@REQUESTDATE"/>
      </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@REQUESTDATE"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
    <xsl:apply-templates select="./OBR/NTE" mode="nte"/>
    <!-- to display Order Comments -->
    <xsl:apply-templates select="*" mode="obr"/>
  </xsl:element>

```

Figure D-10 HL7_OUT.xsl

```

</xsl:template>
<!--
*****
*      OBR/OBX Template      *
*****
-->
<xsl:template match="OBR/OBX" mode="obr">
  <xsl:element name="REC">
    <xsl:attribute name="ID">OBR</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SET" />
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
  <!-- to assign the 2nd position of OBR with Order Nr -->
  <xsl:choose>
    <xsl:when test="$posSID">
      <xsl:element name="FIELD">
        <xsl:attribute name="NO">2</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="../@SAMPLEID" />
        </xsl:attribute>
        <xsl:element name="DATA">
          <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
              <xsl:value-of select="../@SAMPLEID" />
            </xsl:attribute>
          </xsl:element>
        </xsl:element>
      </xsl:when>
    <xsl:otherwise>
      <xsl:element name="FIELD">
        <xsl:attribute name="NO">2</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="../@ORDERNR"/>
        </xsl:attribute>
        <xsl:element name="DATA">
          <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
              <xsl:value-of select="../@ORDERNR"/>
            </xsl:attribute>
          </xsl:element>
        </xsl:element>
      </xsl:otherwise>
    </xsl:choose>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">4</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ID" />
      </xsl:attribute>
    </xsl:element>
  </xsl:choose>

```

Figure D-10 HL7_OUT.xsl

```

    <xsl:value-of select="@ANALYTNR" />
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ID"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ANALYTNR"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">7</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@COMPLETEDATE"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@COMPLETEDATE"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@COMPLETEDATE"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">49</xsl:attribute>
  <xsl:variable name="confidential">
    <xsl:choose>
      <xsl:when test="@CONFIDENTIAL=1">C</xsl:when>
      <xsl:otherwise>N</xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="$confidential"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$confidential"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>

```

Figure D-10 HL7_OUT.xsl

```

        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
            <xsl:value-of select="$confidential"/>
        </xsl:attribute>
    </xsl:element>
</xsl:element>
</xsl:element>
</xsl:element>
</xsl:element>
<!--
<xsl:element name="FIELD">
<xsl:attribute name="NO">25</xsl:attribute>
<xsl:attribute name="TEXT"><xsl:value-of select="../@ID"/></xsl:attribute>
<xsl:element name="DATA">
<xsl:element name="COMP">
<xsl:attribute name="NO">1</xsl:attribute>
<xsl:attribute name="TEXT"><xsl:value-of select="../@ID"/></xsl:attribute>
<xsl:element name="SUB">
<xsl:attribute name="NO">1</xsl:attribute>
<xsl:attribute name="TEXT"><xsl:value-of select="../@ID"/></xsl:attribute>
</xsl:element>
</xsl:element>
</xsl:element>
</xsl:element>
-->
    <xsl:element name="REC">
        <xsl:attribute name="ID">OBX</xsl:attribute>
        <xsl:element name="FIELD">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@SET"/>
            </xsl:attribute>
            <xsl:element name="DATA">
                <xsl:element name="COMP">
                    <xsl:attribute name="NO">1</xsl:attribute>
                    <xsl:attribute name="TEXT">
                        <xsl:value-of select="@SET"/>
                    </xsl:attribute>
                </xsl:element>
            </xsl:element>
        </xsl:element>
    </xsl:element>
<!--
<xsl:element name="FIELD">
<xsl:attribute name="NO">2</xsl:attribute>
<xsl:attribute name="TEXT"><xsl:value-of select="@ANALYTNR"/></xsl:attribute>
<xsl:element name="DATA">
<xsl:element name="COMP">
<xsl:attribute name="NO">1</xsl:attribute>
<xsl:attribute name="TEXT"><xsl:value-of select="@ANALYTNR"/></xsl:attribute>
<xsl:element name="SUB">
<xsl:attribute name="NO">1</xsl:attribute>
<xsl:attribute name="TEXT"><xsl:value-of select="@ANALYTNR"/></xsl:attribute>
</xsl:element>
</xsl:element>
</xsl:element>
</xsl:element>

```

Figure D-10 HL7_OUT.xsl

```

-->
<xsl:element name="FIELD">
  <xsl:attribute name="NO">3</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@ID" />
    ^
    <xsl:value-of select="@ANALYTNR" />
    ^
    <xsl:value-of select="@PROFIL" />
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ID"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@ID"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ANALYTNR"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@ANALYTNR"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@PROFIL"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@PROFIL"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">5</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@VALUE" />
    ^
    <xsl:value-of select="@VALIDSTAT" />

```

Figure D-10 HL7_OUT.xsl

```

</xsl:attribute>
<xsl:element name="DATA">
  <xsl:element name="COMP">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@VALUE"/>
    </xsl:attribute>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@VALUE"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="COMP">
  <xsl:attribute name="NO">2</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@VALIDSTAT"/>
  </xsl:attribute>
  <xsl:element name="SUB">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@VALIDSTAT"/>
    </xsl:attribute>
  </xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">6</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@UNITS"/>
  </xsl:attribute>
</xsl:element>
<xsl:element name="DATA">
  <xsl:element name="COMP">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@UNITS"/>
    </xsl:attribute>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@UNITS"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <!-- reference range - Unormal - Onormal - Verd-->
  <xsl:attribute name="NO">7</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@REFBEREICH" />
  </xsl:attribute>
  ^

```

Figure D-10 HL7_OUT.xsl


```

    <xsl:value-of select="@UNORMAL" />
    ^
    <xsl:value-of select="@ONORMAL" />
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@REFBEREICH"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@REFBEREICH"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@UNORMAL"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@UNORMAL"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ONORMAL"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@ONORMAL"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <!-- Abnormal Flags -->
    <xsl:attribute name="NO">8</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@INDIKATOR"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@INDIKATOR"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>

```

Figure D-10 HL7_OUT.xsl

```

        </xsl:attribute>
        <xsl:element name="SUB">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@INDIKATOR"/>
            </xsl:attribute>
        </xsl:element>
    </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">11</xsl:attribute>
    <xsl:attribute name="TEXT">F</xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">F</xsl:attribute>
            <xsl:element name="SUB">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">F</xsl:attribute>
            </xsl:element>
        </xsl:element>
    </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">14</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@COMPLETEDATE"/>
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@COMPLETEDATE"/>
            </xsl:attribute>
            <xsl:element name="SUB">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">
                    <xsl:value-of select="@COMPLETEDATE"/>
                </xsl:attribute>
            </xsl:element>
        </xsl:element>
    </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">15</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@GERNR" />
        ^
        <xsl:value-of select="@GERNAME" />
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>

```

Figure D-10 HL7_OUT.xsl

```

        <xsl:attribute name="TEXT">
            <xsl:value-of select="@GERNR"/>
        </xsl:attribute>
        <!-- this might not be needed
<xsl:element name="SUB">
    <xsl:attribute name="NO">1</xsl:attribute>
<xsl:attribute name="TEXT">
    <xsl:value-of select="@GERNR" />
</xsl:attribute>
</xsl:element>    -->
        </xsl:element>
        <xsl:element name="COMP">
            <xsl:attribute name="NO">2</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@GERNAME"/>
            </xsl:attribute>
            <!-- this might not be needed
<xsl:element name="SUB">
    <xsl:attribute name="NO">1</xsl:attribute>
<xsl:attribute name="TEXT">
    <xsl:value-of select="@GERNAME" />
</xsl:attribute>
</xsl:element>    -->
        </xsl:element>
        </xsl:element>
        </xsl:element>

<xsl:element name="FIELD">
    <!-- RELEASEUID - RELEASEDATE - MEDVALUID - MEDVALDATE -->
    <xsl:attribute name="NO">16</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@RELEASEUID" />
        ^
        <xsl:value-of select="@RELEASEDATE" />
        ^
        <xsl:value-of select="@MEDVALUID" />
        ^
        <xsl:value-of select="@MEDVALDATE" />
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@RELEASEUID"/>
            </xsl:attribute>
            <xsl:element name="SUB">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">
                    <xsl:value-of select="@RELEASEUID"/>
                </xsl:attribute>
            </xsl:element>
        </xsl:element>
        <xsl:element name="COMP">
            <xsl:attribute name="NO">2</xsl:attribute>
            <xsl:attribute name="TEXT">

```

Figure D-10 HL7_OUT.xsl

```

        <xsl:value-of select="@RELEASEDATE"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@RELEASEDATE"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@MEDVALUID"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@MEDVALUID"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">4</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@MEDVALDATE"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@MEDVALDATE"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:element>

<xsl:element name="FIELD">
  <xsl:attribute name="NO">19</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@INSTRUMENT_MEASUREMENT_TIME"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@INSTRUMENT_MEASUREMENT_TIME"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@INSTRUMENT_MEASUREMENT_TIME"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>

```

Figure D-10 HL7_OUT.xsl

```

    </xsl:element>

    <xsl:apply-templates select="*" mode="nte"/>
  </xsl:element>
</xsl:template>
<!--
*****
*      Enf of OBR Template      *
*****
-->
<!--
*****
*      NTE Template (comment)  *
*****
-->

<xsl:template match="NTE" mode="nte">
  <xsl:element name="REC">
    <xsl:attribute name="ID">NTE</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SET"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@SET"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>

  <xsl:element name="FIELD">
    <xsl:attribute name="NO">2</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@SOURCE"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@SOURCE"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>

  <xsl:choose>
    <xsl:when test="@SOURCE = 'I'">
      <xsl:element name="FIELD">
        <xsl:attribute name="NO">3</xsl:attribute>
        <xsl:attribute name="TEXT">

```

Figure D-10 HL7_OUT.xsl

```

    <xsl:text>Code: </xsl:text>
    <xsl:value-of select="@CODE"/>
    <xsl:text> Comment: </xsl:text>
    <xsl:value-of select="@DESC"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:text>Code: </xsl:text>
        <xsl:value-of select="@CODE"/>
        <xsl:text> Comment: </xsl:text>
        <xsl:value-of select="@DESC"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
</xsl:when>
<xsl:when test="@SOURCE = 'L'">
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">3</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@TEXT"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@TEXT"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:when>
<xsl:otherwise>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">3</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@TEXT"/>
      <xsl:value-of select="@DESC"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@TEXT"/>
          <xsl:value-of select="@DESC"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:otherwise>
</xsl:choose>
</xsl:element>
<!-- end of REC NTE -->

```

Figure D-10 HL7_OUT.xsl

```

</xsl:template>
<!-- *****
*           End of NTE Template (comment)           *
***** -->
<!-- *****
*           footer                                   *
***** -->
<xsl:template match="MSH" mode="footer">
</xsl:template>
<xsl:template match="*" mode="header">
  <xsl:apply-templates select="*" mode="header"/>
</xsl:template>
<xsl:template match="*" mode="body">
  <xsl:apply-templates select="*" mode="body"/>
</xsl:template>
<xsl:template match="*" mode="footer">
  <xsl:apply-templates select="*" mode="footer"/>
</xsl:template>
<!--*****
*END OF DOCUMENT
*****
-->
</xsl:stylesheet>

```

Figure D-10 HL7_OUT.xsl

The HL7 quality control output templates

This section gives the XSLT used to create the HL7 quality control result messages.
This file is HL7-QualityControlOUT.xsl.

```

<?xml version="1.0" encoding="utf-8"?>
<!--
*****
*DOCUMENT ELEMENT: xsl:stylesheet
*****
-->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" indent="yes" version="1.0" encoding="UTF-8"/>
  <!--</xsl:param>-->
  <xsl:param name="SENDING_APPLICATION">
    <!-- <data name="SENDING_APPLICATION" comment="The name of the sending application"
type="string" /> -->
    <xsl:value-of select="'" />
  </xsl:param>
  <xsl:param name="SENDING_FACILITY">
    <!-- <data name="SENDING_FACILITY" comment="The name of the sending facility" type="string" /> -
->
    <xsl:value-of select="'" />
  </xsl:param>
  <xsl:param name="RECEIVING_APPLICATION">
    <!-- <data name="RECEIVING_APPLICATION" comment="The name of the receiving application"
type="string" /> -->
    <xsl:value-of select="'" />
  </xsl:param>
  <xsl:param name="RECEIVING_FACILITY">

```

Figure D-11 HL7-QualityControlOUT.xsl

```

<!-- <data name="RECEIVING_FACILITY" comment="The name of the receiving facility" type="string"
/> -->
<xsl:value-of select="'"'/>
</xsl:param>

<xsl:param name="ACCEPT_TYPE">
<!-- <data name="ACCEPT_TYPE" comment="[MSH 15] The type of acknowledge , AL (Always), NE (Never)
,ER (Error) , SU (Successful)" type="string" /> -->
<xsl:value-of select="'"'/>
</xsl:param>

<xsl:param name="ACK_TYPE">
<!-- <data name="ACK_TYPE" comment="[MSH 16] The type of application acknowledge , AL (Always),
NE (Never) ,ER (Error) , SU (Successful)" type="string" /> -->
<xsl:value-of select="'"'/>
</xsl:param>

<!-- Local variables -->
<xsl:param name="L_SENDING_APPLICATION">
<xsl:choose>
<xsl:when test="string-length($SENDING_APPLICATION) != 0">
<xsl:value-of select="$SENDING_APPLICATION"/>
</xsl:when>
<xsl:otherwise>
<xsl:value-of select="//MSH/@SENDING_APPLICATION"/>
</xsl:otherwise>
</xsl:choose>
</xsl:param>
<xsl:param name="L_SENDING_FACILITY">
<xsl:choose>
<xsl:when test="string-length($SENDING_FACILITY) != 0">
<xsl:value-of select="$SENDING_FACILITY"/>
</xsl:when>
<xsl:otherwise>
<xsl:value-of select="//MSH/@SENDING_FACILITY"/>
</xsl:otherwise>
</xsl:choose>
</xsl:param>
<xsl:param name="L_RECEIVING_APPLICATION">
<xsl:choose>
<xsl:when test="string-length($RECEIVING_APPLICATION) != 0">
<xsl:value-of select="$RECEIVING_APPLICATION"/>
</xsl:when>
<xsl:otherwise>
<xsl:value-of select="//MSH/@RECEIVING_APPLICATION"/>
</xsl:otherwise>
</xsl:choose>
</xsl:param>
<xsl:param name="L_RECEIVING_FACILITY">
<xsl:choose>
<xsl:when test="string-length($RECEIVING_FACILITY) != 0">
<xsl:value-of select="$RECEIVING_FACILITY"/>
</xsl:when>
<xsl:otherwise>
<xsl:value-of select="//MSH/@RECEIVING_FACILITY"/>

```

Figure D-11 HL7-QualityControlOUT.xsl


```

        </xsl:otherwise>
    </xsl:choose>
</xsl:param>
<xsl:param name="L_ACCEPT_TYPE">
    <xsl:choose>
        <xsl:when test="string-length($ACCEPT_TYPE) != 0">
            <xsl:value-of select="$ACCEPT_TYPE"/>
        </xsl:when>
        <xsl:otherwise>
            <xsl:value-of select="//MSH/@ACCEPT_TYPE"/>
        </xsl:otherwise>
    </xsl:choose>
</xsl:param>

<xsl:param name="L_ACK_TYPE">
    <xsl:choose>
        <xsl:when test="string-length($ACK_TYPE) != 0">
            <xsl:value-of select="$ACK_TYPE"/>
        </xsl:when>
        <xsl:otherwise>
            <xsl:value-of select="//MSH/@ACK_TYPE"/>
        </xsl:otherwise>
    </xsl:choose>
</xsl:param>

<xsl:template match="/">
    <xsl:element name="HL7">
        <xsl:attribute name="DIRECTION">OUT</xsl:attribute>
        <xsl:attribute name="MODULE">cobasIT3000</xsl:attribute>
        <xsl:apply-templates select="*" mode="header"/>
        <xsl:apply-templates select="*" mode="body"/>
        <xsl:apply-templates select="*" mode="footer"/>
    </xsl:element>
</xsl:template>
<!-- header -->
<!--
*****
*           MSH Template (header)                               *
*****
-->

<xsl:template match="MSH" mode="header">
    <xsl:element name="REC">
        <xsl:attribute name="ID">MSH</xsl:attribute>
        <xsl:attribute name="TYPE">HEADER</xsl:attribute>
        <xsl:element name="FIELD">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">|</xsl:attribute>
        </xsl:element>
        <xsl:element name="FIELD">
            <xsl:attribute name="NO">2</xsl:attribute>
            <xsl:attribute name="TEXT">^~\&amp;</xsl:attribute>
        </xsl:element>
        <xsl:element name="FIELD">
            <xsl:attribute name="NO">3</xsl:attribute>
            <xsl:attribute name="TEXT">

```

Figure D-11 HL7-QualityControlOUT.xsl

```

    <xsl:value-of select="$L_SENDING_APPLICATION"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_SENDING_APPLICATION"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="$L_SENDING_APPLICATION"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">4</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="$L_SENDING_FACILITY"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="$L_SENDING_FACILITY"/>
        </xsl:attribute>
        <xsl:element name="SUB">
          <xsl:attribute name="NO">1</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="$L_SENDING_FACILITY"/>
          </xsl:attribute>
        </xsl:element>
      </xsl:element>
    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">5</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="$L_RECEIVING_APPLICATION"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="$L_RECEIVING_APPLICATION"/>
        </xsl:attribute>
        <xsl:element name="SUB">
          <xsl:attribute name="NO">1</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="$L_RECEIVING_APPLICATION"/>
          </xsl:attribute>
        </xsl:element>
      </xsl:element>
    </xsl:element>
  </xsl:element>

```

Figure D-11 HL7-QualityControlOUT.xsl

```

</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">6</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="$L_RECEIVING_FACILITY"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_RECEIVING_FACILITY"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_RECEIVING_FACILITY"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">7</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@DATE"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@DATE"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@DATE"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">9</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@MESSAGE_TYPE" />
    ^
    <xsl:value-of select="@EVENT_TYPE" />
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@MESSAGE_TYPE"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>

```

Figure D-11 HL7-QualityControlOUT.xsl

```

    <!--
<xsl:element name="SUB">
  <xsl:attribute name="NO">1</xsl:attribute>
<xsl:attribute name="TEXT">
  <xsl:value-of select="@MESSAGE_TYPE" />
</xsl:attribute>
</xsl:element>
-->

  </xsl:element>
  <xsl:element name="COMP">
    <xsl:attribute name="NO">2</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@EVENT_TYPE" />
    </xsl:attribute>
    <!-- <xsl:element name="SUB">
<xsl:attribute name="NO">2</xsl:attribute>
<xsl:attribute name="TEXT">
  <xsl:value-of select="@EVENT_TYPE" />
</xsl:attribute>
</xsl:element> -->
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">10</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@CONTROL_ID" />
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@CONTROL_ID" />
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@CONTROL_ID" />
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">11</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@PROCESSING_ID" />
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@PROCESSING_ID" />
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>

```

Figure D-11 HL7-QualityControlOUT.xsl

```

        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
            <xsl:value-of select="@PROCESSING_ID"/>
        </xsl:attribute>
    </xsl:element>
</xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">12</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@VERSION"/>
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@VERSION"/>
            </xsl:attribute>
            <xsl:element name="SUB">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">
                    <xsl:value-of select="@VERSION"/>
                </xsl:attribute>
            </xsl:element>
        </xsl:element>
    </xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">15</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_ACCEPT_TYPE"/>
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="$L_ACCEPT_TYPE"/>
            </xsl:attribute>
            <xsl:element name="SUB">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">
                    <xsl:value-of select="$L_ACCEPT_TYPE"/>
                </xsl:attribute>
            </xsl:element>
        </xsl:element>
    </xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">16</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_ACK_TYPE"/>
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">

```

Figure D-11 HL7-QualityControlOUT.xsl

```

        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
            <xsl:value-of select="$L_ACK_TYPE"/>
        </xsl:attribute>
        <xsl:element name="SUB">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="$L_ACK_TYPE"/>
            </xsl:attribute>
        </xsl:element>
    </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">18</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@ENCODING"/>
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@ENCODING"/>
            </xsl:attribute>
            <xsl:element name="SUB">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">
                    <xsl:value-of select="@ENCODING"/>
                </xsl:attribute>
            </xsl:element>
        </xsl:element>
    </xsl:element>
</xsl:element>
    <xsl:apply-templates select="*" mode="header"/>
</xsl:element>
</xsl:template>
<!--
*****
*           Body Template                               *
*****
-->
<xsl:template match="PID" mode="body">
    <xsl:element name="REC">
        <xsl:attribute name="ID">PID</xsl:attribute>
        <xsl:element name="FIELD">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@SET"/>
            </xsl:attribute>
            <xsl:element name="DATA">
                <xsl:element name="COMP">
                    <xsl:attribute name="NO">1</xsl:attribute>
                    <xsl:attribute name="TEXT">
                        <xsl:value-of select="@SET"/>
                    </xsl:attribute>
                </xsl:element>
            </xsl:element>
        </xsl:element>
    </xsl:element>

```

Figure D-11 HL7-QualityControlOUT.xsl

```

    </xsl:element>
  </xsl:element>
</xsl:element>
<!--
<xsl:element name="FIELD">
  <xsl:attribute name="NO">
    <xsl:value-of select="$patid_pos"/>
  </xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@PATID"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@PATID"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">5</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@LASTNAME" />
    ^
    <xsl:value-of select="@FIRSTNAME" />
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@LASTNAME"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@FIRSTNAME"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">7</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@DOB"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@DOB"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>

```

Figure D-11 HL7-QualityControlOUT.xsl

```

</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">8</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@SEX"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SEX"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:apply-templates select="*" mode="pv1"/>
-->
  <xsl:apply-templates select="*" mode="orc"/>
</xsl:element>
</xsl:template>
<!--
*****
*          PV1 Template                      *
*****
-->
<!--
<xsl:template match="PV1" mode="pv1">
  <xsl:element name="REC">
    <xsl:attribute name="ID">PV1</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SET"/>
      </xsl:attribute>
      <xsl:element name="DATA">
        <xsl:element name="COMP">
          <xsl:attribute name="NO">1</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@SET"/>
          </xsl:attribute>
        </xsl:element>
      </xsl:element>
    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">19</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@VISITNR"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@VISITNR"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>

```

Figure D-11 HL7-QualityControlOUT.xsl


```

    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">44</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@ADMITDATE"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@ADMITDATE"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">45</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@DISCHARGEDATE"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@DISCHARGEDATE"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:template>
-->
<xsl:template match="ORC" mode="orc">
  <xsl:element name="REC">
    <xsl:attribute name="ID">ORC</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SET"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@CONTROL_HOST_ID"/>
      </xsl:attribute>
      <xsl:element name="DATA">
        <xsl:element name="COMP">
          <xsl:attribute name="NO">1</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@CONTROL_HOST_ID"/>
          </xsl:attribute>
        </xsl:element>
      </xsl:element>
    </xsl:element>
  </xsl:element>

```

Figure D-11 HL7-QualityControlOUT.xsl

```

</xsl:element>
<!-- LIKE PSM field 5 is 'IP' -->
<xsl:element name="FIELD">
  <xsl:attribute name="NO">5</xsl:attribute>
  <xsl:attribute name="TEXT">IP</xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">IP</xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<!--routine-->
<xsl:element name="FIELD">
  <xsl:attribute name="NO">7</xsl:attribute>
  <xsl:attribute name="TEXT">^^^^R</xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">6</xsl:attribute>
      <xsl:attribute name="TEXT">R</xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<!--empty-->
<xsl:element name="FIELD">
  <xsl:attribute name="NO">8</xsl:attribute>
  <xsl:attribute name="TEXT"></xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT"></xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>

  <xsl:apply-templates select="*" mode="obr"/>
</xsl:element>
</xsl:template>
<!--
*****
*       OBR/OBX Template       *
*****
-->
<xsl:template match="OBR/OBX" mode="obr">
  <xsl:element name="REC">
    <xsl:attribute name="ID">OBR</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="../@SET"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">15</xsl:attribute>
      <xsl:attribute name="TEXT">

```

Figure D-11 HL7-QualityControlOUT.xsl

```

    <xsl:value-of select="../@SPECIMEN"/>^^^^^Q
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="../@SPECIMEN"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">7</xsl:attribute>
      <xsl:attribute name="TEXT">Q</xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<!-- Instrument Code placed in Filler Field 1-->
<xsl:element name="FIELD">
  <xsl:attribute name="NO">20</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="../@INSTRUMENT"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="../@INSTRUMENT"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">27</xsl:attribute>
  <xsl:attribute name="TEXT">^^^^^R</xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">6</xsl:attribute>
      <xsl:attribute name="TEXT">R</xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<!--

<xsl:element name="FIELD">
  <xsl:attribute name="NO">2</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="../@ORDERNR"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="../@ORDERNR"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>

```

Figure D-11 HL7-QualityControlOUT.xsl

```

</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">4</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@ID"/>^<xsl:value-of select="@ANALYTNR"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ID"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ANALYTNR"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">7</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@COMPLETEDATE"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@COMPLETEDATE"/>
      </xsl:attribute>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@COMPLETEDATE"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
</xsl:element>

<xsl:element name="FIELD">
  <xsl:attribute name="NO">25</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="../@ID"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="../@ID"/>
      </xsl:attribute>
    <xsl:element name="SUB">

```

Figure D-11 HL7-QualityControlOUT.xsl

```

        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
            <xsl:value-of select="../@ID"/>
        </xsl:attribute>
    </xsl:element>
</xsl:element>
</xsl:element>
</xsl:element>
-->

<xsl:element name="REC">
    <xsl:attribute name="ID">OBX</xsl:attribute>
    <xsl:element name="FIELD">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
            <xsl:value-of select="../@SET"/>
        </xsl:attribute>
        <xsl:element name="DATA">
            <xsl:element name="COMP">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">
                    <xsl:value-of select="../@SET"/>
                </xsl:attribute>
            </xsl:element>
        </xsl:element>
    </xsl:element>
</xsl:element>
<!--
<xsl:element name="FIELD">
<xsl:attribute name="NO">2</xsl:attribute>
<xsl:attribute name="TEXT"><xsl:value-of select="@ANALYTNR"/></xsl:attribute>
<xsl:element name="DATA">
<xsl:element name="COMP">
<xsl:attribute name="NO">1</xsl:attribute>
<xsl:attribute name="TEXT"><xsl:value-of select="@ANALYTNR"/></xsl:attribute>
<xsl:element name="SUB">
<xsl:attribute name="NO">1</xsl:attribute>
<xsl:attribute name="TEXT"><xsl:value-of select="@ANALYTNR"/></xsl:attribute>
</xsl:element>
</xsl:element>
</xsl:element>
</xsl:element>
-->
<xsl:element name="FIELD">
    <xsl:attribute name="NO">3</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@ID" />
        ^
        <xsl:value-of select="@ANALYTNR" />
        ^
        <xsl:value-of select="@PROFIL" />
    </xsl:attribute>
    <xsl:element name="DATA">

```

Figure D-11 HL7-QualityControlOUT.xsl

```

<xsl:element name="COMP">
  <xsl:attribute name="NO">1</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@ID"/>
  </xsl:attribute>
  <xsl:element name="SUB">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@ID"/>
    </xsl:attribute>
  </xsl:element>
</xsl:element>
<xsl:element name="COMP">
  <xsl:attribute name="NO">2</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@ANALYTNR"/>
  </xsl:attribute>
  <xsl:element name="SUB">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@ANALYTNR"/>
    </xsl:attribute>
  </xsl:element>
</xsl:element>
<xsl:element name="COMP">
  <xsl:attribute name="NO">3</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@PROFIL"/>
  </xsl:attribute>
  <xsl:element name="SUB">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@PROFIL"/>
    </xsl:attribute>
  </xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">5</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@VALUE"/>
    ^
    <xsl:value-of select="@VALIDSTAT"/>
  </xsl:attribute>
<xsl:element name="DATA">
  <xsl:element name="COMP">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@VALUE"/>
    </xsl:attribute>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@VALUE"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>

```

Figure D-11 HL7-QualityControlOUT.xsl

```

        </xsl:attribute>
      </xsl:element>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@VALIDSTAT"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@VALIDSTAT"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">6</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@UNITS"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@UNITS"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@UNITS"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <!-- reference range - Unormal - Onormal - Verd-->
  <xsl:attribute name="NO">7</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@REFBEREICH"/>
    ^
    <xsl:value-of select="@UNORMAL"/>
    ^
    <xsl:value-of select="@ONORMAL"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@REFBEREICH"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>

```

Figure D-11 HL7-QualityControlOUT.xsl

```

        <xsl:attribute name="TEXT">
            <xsl:value-of select="@REFBEREICH"/>
        </xsl:attribute>
    </xsl:element>
</xsl:element>
<xsl:element name="COMP">
    <xsl:attribute name="NO">2</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@UNORMAL"/>
    </xsl:attribute>
    <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
            <xsl:value-of select="@UNORMAL"/>
        </xsl:attribute>
    </xsl:element>
</xsl:element>
<xsl:element name="COMP">
    <xsl:attribute name="NO">3</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@ONORMAL"/>
    </xsl:attribute>
    <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
            <xsl:value-of select="@ONORMAL"/>
        </xsl:attribute>
    </xsl:element>
</xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <!-- Abnormal Flags -->
    <xsl:attribute name="NO">8</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@INDIKATOR"/>
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@INDIKATOR"/>
            </xsl:attribute>
            <xsl:element name="SUB">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">
                    <xsl:value-of select="@INDIKATOR"/>
                </xsl:attribute>
            </xsl:element>
        </xsl:element>
    </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">11</xsl:attribute>
    <xsl:attribute name="TEXT">F</xsl:attribute>

```

Figure D-11 HL7-QualityControlOUT.xsl


```

    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">F</xsl:attribute>
        <xsl:element name="SUB">
          <xsl:attribute name="NO">1</xsl:attribute>
          <xsl:attribute name="TEXT">F</xsl:attribute>
        </xsl:element>
      </xsl:element>
    </xsl:element>
  </xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">14</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@COMPLETEDATE"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@COMPLETEDATE"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@COMPLETEDATE"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:element>

<xsl:element name="FIELD">
  <xsl:attribute name="NO">15</xsl:attribute>
  <xsl:attribute name="TEXT">
    ^^^
    <xsl:value-of select="../@INSTRUMENT"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="../@INSTRUMENT"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@GERNAME"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>

<xsl:element name="FIELD">

```

Figure D-11 HL7-QualityControlOUT.xsl

```

    <xsl:attribute name="NO">18</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="../@INSTRUMENT"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="../@INSTRUMENT"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>

  <xsl:element name="FIELD">
    <xsl:attribute name="NO">19</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@INSTRUMENT_MEASUREMENT_TIME"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@INSTRUMENT_MEASUREMENT_TIME"/>
        </xsl:attribute>
        <xsl:element name="SUB">
          <xsl:attribute name="NO">1</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@INSTRUMENT_MEASUREMENT_TIME"/>
          </xsl:attribute>
        </xsl:element>
      </xsl:element>
    </xsl:element>
  </xsl:element>

  <xsl:apply-templates select="*" mode="nte"/>
</xsl:element>
</xsl:template>
</xsl:template>
<!--
*****
*      Enf of OBR Template      *
*****
-->
<!--
*****
*      NTE Template (comment)   *
*****
-->
<xsl:template match="NTE" mode="nte">
  <xsl:element name="REC">
    <xsl:attribute name="ID">NTE</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">1</xsl:attribute>

```

Figure D-11 HL7-QualityControlOUT.xsl

```

<xsl:attribute name="TEXT">
  <xsl:value-of select="@SET"/>
</xsl:attribute>
<xsl:element name="DATA">
  <xsl:element name="COMP">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@SET"/>
    </xsl:attribute>
  </xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">2</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@SOURCE"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SOURCE"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:choose>
  <xsl:when test="@SOURCE = 'I'">
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:text>Code:</xsl:text>
        <xsl:value-of select="@CODE"/>
        <xsl:text> Comment:</xsl:text>
        <xsl:value-of select="@DESC"/>
      </xsl:attribute>
      <xsl:element name="DATA">
        <xsl:element name="COMP">
          <xsl:attribute name="NO">1</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:text>Code:</xsl:text>
            <xsl:value-of select="@CODE"/>
            <xsl:text> Comment:</xsl:text>
            <xsl:value-of select="@DESC"/>
          </xsl:attribute>
        </xsl:element>
      </xsl:element>
    </xsl:when>
    <xsl:when test="@SOURCE = 'L'">
      <xsl:element name="FIELD">
        <xsl:attribute name="NO">3</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@TEXT"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:when>
  </xsl:choose>

```

Figure D-11 HL7-QualityControlOUT.xsl

```

        <xsl:element name="DATA">
            <xsl:element name="COMP">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">
                    <xsl:value-of select="@TEXT"/>
                </xsl:attribute>
            </xsl:element>
        </xsl:element>
    </xsl:when>
    <xsl:otherwise>
        <xsl:element name="FIELD">
            <xsl:attribute name="NO">3</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@TEXT"/>
                <xsl:value-of select="@DESC"/>
            </xsl:attribute>
            <xsl:element name="DATA">
                <xsl:element name="COMP">
                    <xsl:attribute name="NO">1</xsl:attribute>
                    <xsl:attribute name="TEXT">
                        <xsl:value-of select="@TEXT"/>
                        <xsl:value-of select="@DESC"/>
                    </xsl:attribute>
                </xsl:element>
            </xsl:element>
        </xsl:otherwise>
    </xsl:choose>
</xsl:element>
<!-- end of REC NTE -->
</xsl:template>
<!-- *****
*           End of NTE Template (comment)           *
***** -->
<!-- *****
*           footer                                   *
***** -->
<xsl:template match="MSH" mode="footer">
</xsl:template>
<xsl:template match="*" mode="header">
    <xsl:apply-templates select="*" mode="header"/>
</xsl:template>
<xsl:template match="*" mode="body">
    <xsl:apply-templates select="*" mode="body"/>
</xsl:template>
<xsl:template match="*" mode="footer">
    <xsl:apply-templates select="*" mode="footer"/>
</xsl:template>
<!--*****
*END OF DOCUMENT
*****
-->
</xsl:stylesheet>

```

Figure D-11 HL7-QualityControlOUT.xsl

The HL7 sample event templates

This section gives the XSLT that creates the HL7 sample event messages sent to the host. This file is SampleSeen_Hl7.xsl.

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" indent="yes" version="1.0" encoding="UTF-8"/>
  <xsl:param name="SENDING_APPLICATION">
    <!-- <data name="SENDING_APPLICATION" comment="The name of the sending application"
type="string" /> -->
    <xsl:value-of select="'"'/>
  </xsl:param>
  <xsl:param name="SENDING_FACILITY">
    <!-- <data name="SENDING_FACILITY" comment="The name of the sending facility" type="string" /> -
->
    <xsl:value-of select="'"'/>
  </xsl:param>
  <xsl:param name="RECEIVING_APPLICATION">
    <!-- <data name="RECEIVING_APPLICATION" comment="The name of the receiving application"
type="string" /> -->
    <xsl:value-of select="'"'/>
  </xsl:param>
  <xsl:param name="RECEIVING_FACILITY">
    <!-- <data name="RECEIVING_FACILITY" comment="The name of the receiving facility" type="string"
/> -->
    <xsl:value-of select="'"'/>
  </xsl:param>
  <xsl:param name="SEND_ONCE">
    <!-- <data name="SEND_ONCE" type="string" comment="List of events for sending messages only once"
/> -->
    <xsl:value-of select="'"'/>
  </xsl:param>
  <xsl:param name="SEND_INSTRUMENTS">
    <!-- <data name="INSTRUMENTS" type="string" comment="List of instruments" /> -->
    <xsl:value-of select="ALL"/>
  </xsl:param>
  <xsl:param name="SEND_LOCATIONS">
    <!-- <data name="Locations" type="string" comment="List of locations" /> -->
    <xsl:value-of select="ALL"/>
  </xsl:param>
  <xsl:param name="ACCEPT_TYPE">
    <!-- <data name="ACCEPT_TYPE" comment="[MSH 15] The type of acknowledge , AL (Always), NE (Never)
,ER (Error) , SU (Successful)" type="string" /> -->
    <xsl:value-of select="'"'/>
  </xsl:param>
  <xsl:param name="ACK_TYPE">
    <!-- <data name="ACK_TYPE" comment="[MSH 16] The type of application acknowledge , AL (Always),
NE (Never) ,ER (Error) , SU (Successful)" type="string" /> -->
    <xsl:value-of select="'"'/>
  </xsl:param>
  <!-- Local variables -->
  <xsl:param name="L_SENDING_APPLICATION">
    <xsl:choose>
      <xsl:when test="string-length($SENDING_APPLICATION) != 0">
```

Figure D-12 SampleSeen_Hl7.xsl

```

        <xsl:value-of select="$SENDING_APPLICATION"/>
    </xsl:when>
    <xsl:otherwise>
        <xsl:value-of select="/SampleEvent/MSH/@SENDING_APPLICATION"/>
    </xsl:otherwise>
    </xsl:choose>
</xsl:param>
<xsl:param name="L_SENDING_FACILITY">
    <xsl:choose>
        <xsl:when test="string-length($SENDING_FACILITY) != 0">
            <xsl:value-of select="$SENDING_FACILITY"/>
        </xsl:when>
        <xsl:otherwise>
            <xsl:value-of select="/SampleEvent/MSH/@SENDING_FACILITY"/>
        </xsl:otherwise>
    </xsl:choose>
</xsl:param>
<xsl:param name="L_RECEIVING_APPLICATION">
    <xsl:choose>
        <xsl:when test="string-length($RECEIVING_APPLICATION) != 0">
            <xsl:value-of select="$RECEIVING_APPLICATION"/>
        </xsl:when>
        <xsl:otherwise>
            <xsl:value-of select="/SampleEvent/MSH/@RECEIVING_APPLICATION"/>
        </xsl:otherwise>
    </xsl:choose>
</xsl:param>
<xsl:param name="L_RECEIVING_FACILITY">
    <xsl:choose>
        <xsl:when test="string-length($RECEIVING_FACILITY) != 0">
            <xsl:value-of select="$RECEIVING_FACILITY"/>
        </xsl:when>
        <xsl:otherwise>
            <xsl:value-of select="/SampleEvent/MSH/@RECEIVING_FACILITY"/>
        </xsl:otherwise>
    </xsl:choose>
</xsl:param>
<xsl:param name="L_ACCEPT_TYPE">
    <xsl:choose>
        <xsl:when test="string-length($ACCEPT_TYPE) != 0">
            <xsl:value-of select="$ACCEPT_TYPE"/>
        </xsl:when>
        <xsl:otherwise>
            <xsl:value-of select="//MSH/@ACCEPT_TYPE"/>
        </xsl:otherwise>
    </xsl:choose>
</xsl:param>
<xsl:param name="L_ACK_TYPE">
    <xsl:choose>
        <xsl:when test="string-length($ACK_TYPE) != 0">
            <xsl:value-of select="$ACK_TYPE"/>
        </xsl:when>
        <xsl:otherwise>
            <xsl:value-of select="//MSH/@ACK_TYPE"/>
        </xsl:otherwise>
    </xsl:choose>
</xsl:param>

```

Figure D-12 SampleSeen_HI7.xsl

```

    </xsl:otherwise>
  </xsl:choose>
</xsl:param>

<xsl:template match="/">
  <xsl:element name="HL7">
    <xsl:attribute name="DIRECTION">OUT</xsl:attribute>
    <xsl:attribute name="MODULE">LIS</xsl:attribute>
    <xsl:attribute name="SOURCE">CIT5K-DB</xsl:attribute>
    <xsl:attribute name="DESTINATION">CIT5K-HL7</xsl:attribute>
    <xsl:apply-templates select="/SampleEvent/MSH" mode="header"/>
    <xsl:apply-templates select="/SampleEvent/EQU" mode="equ"/>
    <xsl:apply-templates select="/SampleEvent/EQU/PID/SAC" mode="sac"/>
  </xsl:element>
</xsl:template>

<xsl:template match="MSH" mode="header">
  <xsl:element name="REC">
    <xsl:attribute name="ID">MSH</xsl:attribute>
    <xsl:attribute name="TYPE">HEADER</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@FORMAT"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_SENDING_APPLICATION"/>
      </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="$L_SENDING_APPLICATION"/>
        </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="$L_SENDING_APPLICATION"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">4</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="$L_SENDING_FACILITY"/>
    </xsl:attribute>
    <xsl:element name="DATA">
      <xsl:element name="COMP">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="$L_SENDING_FACILITY"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:template>

```

Figure D-12 SampleSeen_HL7.xsl

```

        </xsl:attribute>
        <xsl:element name="SUB">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="$L_SENDING_FACILITY"/>
            </xsl:attribute>
        </xsl:element>
    </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">5</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_RECEIVING_APPLICATION"/>
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="$L_RECEIVING_APPLICATION"/>
            </xsl:attribute>
            <xsl:element name="SUB">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">
                    <xsl:value-of select="$L_RECEIVING_APPLICATION"/>
                </xsl:attribute>
            </xsl:element>
        </xsl:element>
    </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">6</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_RECEIVING_FACILITY"/>
    </xsl:attribute>
    <xsl:element name="DATA">
        <xsl:element name="COMP">
            <xsl:attribute name="NO">1</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="$L_RECEIVING_FACILITY"/>
            </xsl:attribute>
            <xsl:element name="SUB">
                <xsl:attribute name="NO">1</xsl:attribute>
                <xsl:attribute name="TEXT">
                    <xsl:value-of select="$L_RECEIVING_FACILITY"/>
                </xsl:attribute>
            </xsl:element>
        </xsl:element>
    </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
    <xsl:attribute name="NO">7</xsl:attribute>
    <xsl:attribute name="TEXT">
        <xsl:value-of select="@DATE"/>
    </xsl:attribute>

```

Figure D-12 SampleSeen_Hl7.xsl


```

</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">9</xsl:attribute>
  <DATA>
    <COMP>
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">SSU</xsl:attribute>
    </COMP>
    <COMP>
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">U03</xsl:attribute>
    </COMP>
  </DATA>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">10</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@CONTROL_ID"/>
  </xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">11</xsl:attribute>
  <xsl:attribute name="TEXT">P</xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">12</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@VERSION"/>
  </xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">15</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="$L_ACCEPT_TYPE"/>
  </xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_ACCEPT_TYPE"/>
      </xsl:attribute>
      <xsl:element name="SUB">
        <xsl:attribute name="NO">1</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="$L_ACCEPT_TYPE"/>
        </xsl:attribute>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">16</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="$L_ACK_TYPE"/>
  </xsl:attribute>

```

Figure D-12 SampleSeen_Hl7.xsl

```

<xsl:element name="DATA">
  <xsl:element name="COMP">
    <xsl:attribute name="NO">1</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="$L_ACK_TYPE"/>
    </xsl:attribute>
    <xsl:element name="SUB">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_ACK_TYPE"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:element>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">18</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@ENCODING"/>
  </xsl:attribute>
</xsl:element>
</xsl:element>
</xsl:template>
<xsl:template match="EQU" mode="equ">
  <xsl:element name="REC">
    <xsl:attribute name="ID">EQU</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">1</xsl:attribute>
      <DATA>
        <COMP>
          <xsl:attribute name="NO">4</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@EQUIPMENT_IDENTIFIER_NAME"/>
          </xsl:attribute>
        </COMP>
        <COMP>
          <xsl:attribute name="NO">5</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@EQUIPMENT_IDENTIFIER_CLIENT"/>
          </xsl:attribute>
        </COMP>
        <COMP>
          <xsl:attribute name="NO">6</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@EQUIPMENT_IDENTIFIER_MODULE"/>
          </xsl:attribute>
        </COMP>
        <COMP>
          <xsl:attribute name="NO">7</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@EQUIPMENT_IDENTIFIER_EVENT"/>
          </xsl:attribute>
        </COMP>
      </DATA>
    </xsl:element>
  </xsl:element>

```

Figure D-12 SampleSeen_Hl7.xsl

```

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@EVENT_DATE_TIME"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:template>
<xsl:template match="SAC" mode="sac">
  <xsl:element name="REC">
    <xsl:attribute name="ID">SAC</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@CONTAINER_IDENTIFIER"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">4</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@PRIMARY_CONTAINER_IDENTIFIER"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">6</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SPECIMEN_SOURCE"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">7</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="../../MSH/@DATE"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">8</xsl:attribute>
      <DATA>
        <COMP>
          <xsl:attribute name="NO">1</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@CONTAINER_STATUS"/>
          </xsl:attribute>
        </COMP>
        <COMP>
          <xsl:attribute name="NO">4</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@SPECIFIC_CONTAINER_STATUS"/>
          </xsl:attribute>
        </COMP>
      </DATA>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">9</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@CARRIER_TYPE"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>

```

Figure D-12 SampleSeen_Hl7.xsl

```
</xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">10</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@CARRIER_IDENTIFIER"/>
  </xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">11</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@POSITION_IN_CARRIER"/>
  </xsl:attribute>
</xsl:element>
</xsl:element>
</xsl:template>
</xsl:stylesheet>
```

Figure D-12 SampleSeen_HL7.xsl

The default ASTM templates

This section describes the default ASTM templates that **cobas IT 3000** uses for host communication.

The ASTM input template

This section gives the template that interprets incoming ASTM message that the host sends to **cobas IT 3000**. This file is ASTM_IN.xsl.

You cannot send orders containing results

Although the default templates contain fields for orders containing results, this is not supported in the current release. This may be supported in future releases. If you require this functionality, please contact Roche Diagnostics Global Customer support.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!--
*****
* DOCUMENT ELEMENT: xsl:stylesheet
*****
-->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <!-- Get rid of blank lines in output file -->
  <xsl:strip-space elements="*" />

  <xsl:output method="xml" indent="yes" version="1.0" encoding="UTF-8" />

  <xsl:param name="acks">
    <!-- <data name="acks" type="boolean" comment="Use Ack info" /> -->
    <xsl:value-of select="false()" />
  </xsl:param>

  <xsl:param name="patid">
    <!-- <data name="patid" type="integer" comment="Position of patientid within P segment (3,4,5)" /> -->
    <xsl:value-of select="4" />
  </xsl:param>

  <xsl:param name="sampleid">
    <!-- <data name="sampleid" type="integer" comment="Position of sampleid within O segment (3,4)" /> -->
    <xsl:value-of select="4" />
  </xsl:param>

  <xsl:param name="ordernr">
    <!-- <data name="ordernr" type="integer" comment="Position of ordernr within O segment (3,4)" /> -->
    <xsl:value-of select="3" />
  </xsl:param>

  <xsl:param name="resultflags">
```

Figure D-13 ASTM_IN.xsl

Roche Diagnostics

```

<!-- <data name="resultflags" type="string" comment="Result handling flags (Default: 111)" />
-->
  <xsl:value-of select="111"/>
</xsl:param>

<xsl:template match="/">
  <xsl:element name="OrderRequest">
    <xsl:apply-templates select="*" mode="header"/>
    <xsl:apply-templates select="*" mode="body"/>
  </xsl:element>
</xsl:template>

<!-- header -->
<xsl:template match="ASTM/REC[@ID='H']" mode="header">
  <xsl:element name="MSH">
    <xsl:attribute name="SENDER">HOST 3</xsl:attribute>
    <xsl:if test="$acks">
      <xsl:attribute name="MESSAGE_TYPE">MESSAGE_TYPE</xsl:attribute>
      <xsl:attribute name="ACCEPT_TYPE">AL</xsl:attribute>
      <xsl:attribute name="CONTROL_ID">999</xsl:attribute>
    </xsl:if>
  </xsl:element>
</xsl:template>

<!-- body -->
<xsl:template match="REC[@ID='P']" mode="body">
  <xsl:element name="PID">
    <xsl:attribute name="PATID">
      <xsl:value-of select="FIELD[@NO=$patid]/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="LASTNAME">
      <xsl:value-of select="FIELD[@NO='6']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="FIRSTNAME">
      <xsl:value-of select="FIELD[@NO='6']/DATA/COMP[@NO='2']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="NAMEX">
      <xsl:value-of select="FIELD[@NO='6']/DATA/COMP[@NO='3']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="TITLE">
      <xsl:value-of select="FIELD[@NO='6']/DATA/COMP[@NO='5']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="DOB">
      <xsl:value-of select="FIELD[@NO='8']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>
    <xsl:attribute name="SEX">
      <xsl:variable name="sex" select="FIELD[@NO='9']/DATA/COMP[@NO='1']/@TEXT"/>
      <xsl:choose>
        <xsl:when test="$sex='F'">W</xsl:when>
        <xsl:when test="$sex='W'">W</xsl:when>
        <xsl:when test="$sex='M'">M</xsl:when>
        <xsl:otherwise>
          <xsl:value-of select="U"/>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:attribute>
  </xsl:element>

```

Figure D-13 ASTM_IN.xsl

```

</xsl:attribute>
<xsl:attribute name="STREET">
  <xsl:value-of select="FIELD[@NO='11']/DATA/COMP[@NO='1']/@TEXT"/>
</xsl:attribute>
<xsl:attribute name="CITY">
  <xsl:value-of select="FIELD[@NO='11']/DATA/COMP[@NO='2']/@TEXT"/>
</xsl:attribute>
<xsl:attribute name="POSTCODE">
  <xsl:value-of select="FIELD[@NO='11']/DATA/COMP[@NO='4']/@TEXT"/>
</xsl:attribute>
<xsl:attribute name="INSURER">
  <xsl:value-of select="FIELD[@NO='12']/DATA/COMP[@NO='1']/@TEXT"/>
</xsl:attribute>

  <xsl:apply-templates select="*" mode="orc"/>

</xsl:element>
</xsl:template>

<xsl:template match="REC[@ID='O']" mode="orc">
  <xsl:element name="ORC">

    <xsl:attribute name="ORDERNR">
      <!-- if not splitting sample ID into ORDERNR
        <xsl:value-of select="substring(FIELD[@NO=$ordernr]/DATA/COMP[@NO='1']/@TEXT"/>
      -->
      <xsl:value-of select="substring(FIELD[@NO=$ordernr]/DATA/COMP[@NO='1']/@TEXT , 1 ,10)"/>
    </xsl:attribute>

    <xsl:attribute name="SAMPLEID">
      <xsl:value-of select="FIELD[@NO=$sampleid]/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>

    <xsl:attribute name="PRIORITY">
      <xsl:value-of select="FIELD[@NO='6']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>

    <xsl:attribute name="SPECIMEN">
      <xsl:value-of select="FIELD[@NO='16']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>

    <xsl:attribute name="ORDERER">
      <xsl:value-of select="FIELD[@NO='17']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>

    <xsl:attribute name="ORDERDATE">
      <xsl:value-of select="FIELD[@NO='7']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>

    <xsl:attribute name="ORDERDATEFMT">
      <!-- can be given directly -->
      <!-- <xsl:value-of select="FIELD[@NO='7']/DATA/COMP[@NO='1']/@TEXT"/> -->
    </xsl:attribute>

    <xsl:attribute name="WITHDRAWDATE">

```

Figure D-13 ASTM_IN.xsl

```

<!-- can be given directly -->
<xsl:value-of select="FIELD[@NO='8']/DATA/COMP[@NO='1']/@TEXT"/>
</xsl:attribute>

<xsl:attribute name="WITHDRAWDATEFMT">
  <!-- can be given directly -->
  <!-- <xsl:value-of select="FIELD[@NO='8']/DATA/COMP[@NO='1']/@TEXT"/> -->
</xsl:attribute>

<xsl:attribute name="STATUS">
  <!-- laboratory field no 1 -->
  <xsl:variable name="status" select="FIELD[@NO='21']/DATA/COMP[@NO='1']/@TEXT"/>
  <xsl:choose>
    <!-- Some, but not all, results are available -->
    <xsl:when test="$status='A'">A</xsl:when>
    <!-- Order was cancelled -->
    <xsl:when test="$status='CA'">CA</xsl:when>
    <!-- In process, scheduled -->
    <xsl:when test="$status='SC'">SC</xsl:when>
    <!-- Order has been replaced -->
    <xsl:when test="$status='R'">RP</xsl:when>
    <xsl:when test="$status='RP'">RP</xsl:when>
    <!-- Update Order according to HIS_UPDATE_DELETE_LEVEL -->
    <xsl:when test="$status='U'">UP</xsl:when>

    <xsl:otherwise>
      <xsl:value-of select="FIELD[@NO='21']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:otherwise>
    <!-- Default empty -->
  </xsl:choose>
</xsl:attribute>

<xsl:attribute name="CONTROL">
  <xsl:variable name="control" select="FIELD[@NO='26']/DATA/COMP[@NO='1']/@TEXT"/>
  <xsl:choose>
    <!-- New Order -->
    <xsl:when test="$control='O'">NW</xsl:when>
    <!-- Changed Order -->
    <xsl:when test="$control='C'">XO</xsl:when>
    <!-- Canceled Order 06-09-08/alo: chandeg from NW -->
    <xsl:when test="$control='X'">CA</xsl:when>
    <!-- Default New Order -->
    <xsl:otherwise>NW</xsl:otherwise>
  </xsl:choose>
</xsl:attribute>

<xsl:apply-templates select="*" mode="NTE"/>

<!-- Start: OBR -->
<xsl:for-each select="FIELD[@NO='5']/DATA/COMP[@NO='4']">
  <xsl:element name="OBR">
    <xsl:variable name="Testcode" select="@TEXT"/>

    <xsl:attribute name="ACTION">

```

Figure D-13 ASTM_IN.xsl


```

        <xsl:variable name="action" select="../../../FIELD[@NO='12']/DATA/COMP[@NO='1']/
@TEXT"/>

        <xsl:choose>
            <xsl:when test="$action='A'">A</xsl:when>
            <!-- Add -->
            <xsl:when test="$action='C'">C</xsl:when>
            <!-- Cancel -->
            <xsl:when test="$action='N'">C</xsl:when>
            <!-- Cancel -->
            <!--Test action = R for deletion of tests = works -->
            <xsl:when test="$action='R'">R</xsl:when>
            <!-- Test Action code G for Repeats -->
            <xsl:when test="$action='G'">G</xsl:when>
            <xsl:otherwise>
                <xsl:value-of select="$action"/>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:attribute>

    <xsl:attribute name="TESTCODE">
        <xsl:value-of select="$Testcode"/>
    </xsl:attribute>

    <xsl:apply-templates select="../../../REC[@ID='R']" mode="OBX">
        <xsl:with-param name="Testcode" select="$Testcode"/>
    </xsl:apply-templates>

</xsl:element>
</xsl:for-each>
<!-- End: OBR -->

</xsl:element>
</xsl:template>

<!--
*****
*   Add OBX to OBR
*****
-->

    <xsl:template match="*" mode="OBX">
        <xsl:param name="Testcode"/>

        <xsl:variable name="TestcodeOBX">
            <xsl:value-of select="../FIELD[@NO='3']/DATA/COMP[@NO='4']/@TEXT"/>
        </xsl:variable>

        <!-- The ASTM standard stylesheet does not support results for profiles. -->
        <!-- If a repeat delimiter is used in the O record it is very complexe -->
        <!-- to map the correct R record to the generated OBX of the profile -->
        <!-- If it is necessary to add a result to a profile you have to generate -->
        <!-- the following structure for the backend: -->
        <!-- <OBR TESTCODE="YourProfileCode" ...> -->
        <!--     <OBX Testcode="AnalyteCode".../> -->
        <!-- </OBR> -->

```

Figure D-13 ASTM_IN.xsl

```

<!-- The backend will handle all OBX childs of an OBR and will use the -->
<!-- testcode of the OBX to insert the result -->
<xsl:if test="$TestcodeOBX = $Testcode">
  <xsl:element name="OBX">

    <xsl:attribute name="TESTCODE">
      <xsl:value-of select="$TestcodeOBX"/>
    </xsl:attribute>

    <xsl:attribute name="RESULT">
      <xsl:value-of select="FIELD[@NO='4']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>

    <xsl:attribute name="UNIT">
      <xsl:value-of select="FIELD[@NO='5']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>

    <xsl:attribute name="REFRANGE">
      <xsl:value-of select="FIELD[@NO='6']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>

    <xsl:attribute name="INDICATOR">
      <xsl:value-of select="FIELD[@NO='7']/DATA/COMP[@NO='1']/@TEXT"/>
    </xsl:attribute>

    <xsl:attribute name="RESULTFLAGS">
      <xsl:value-of select="$resultflags"/>
    </xsl:attribute>

    <xsl:attribute name="STATUS">
      <xsl:choose>
        <!-- fill in your conditons here as xsl:when -->
        <!-- R: result is already validated and is NOT validated internally, -->
        <!-- validation status 'Single Order' is set -->
        <!-- X: deprecated, LIS_APP offers other possibilities to delete a -->
        <!-- test -->
        <!-- anything else: result is internally validated. It is recommended -->
        <!-- to use the test status '3', other values may have side effects -->
        <xsl:when test="1=1">R</xsl:when>
        <xsl:otherwise>R</xsl:otherwise>
      </xsl:choose>
    </xsl:attribute>

  </xsl:element>
</xsl:if>
</xsl:template>

<!--
*****
* Add order comments C1 .. C5
*****
-->
<xsl:template match="REC[@ID='C']" mode="NTE">
  <xsl:if test="FIELD[@NO='4']/DATA/COMP[@NO='1']/@TEXT">

```

Figure D-13 ASTM_IN.xsl

```

    <xsl:element name="NTE">
      <xsl:attribute name="TEXT">
        <xsl:value-of select="FIELD[@NO='4']/DATA/COMP[@NO='1']/@TEXT"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:if>
  <xsl:if test="FIELD[@NO='4']/DATA/COMP[@NO='2']/@TEXT">
    <xsl:element name="NTE">
      <xsl:attribute name="TEXT">
        <xsl:value-of select="FIELD[@NO='4']/DATA/COMP[@NO='2']/@TEXT"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:if>
  <xsl:if test="FIELD[@NO='4']/DATA/COMP[@NO='3']/@TEXT">
    <xsl:element name="NTE">
      <xsl:attribute name="TEXT">
        <xsl:value-of select="FIELD[@NO='4']/DATA/COMP[@NO='3']/@TEXT"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:if>
  <xsl:if test="FIELD[@NO='4']/DATA/COMP[@NO='4']/@TEXT">
    <xsl:element name="NTE">
      <xsl:attribute name="TEXT">
        <xsl:value-of select="FIELD[@NO='4']/DATA/COMP[@NO='4']/@TEXT"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:if>
  <xsl:if test="FIELD[@NO='4']/DATA/COMP[@NO='5']/@TEXT">
    <xsl:element name="NTE">
      <xsl:attribute name="TEXT">
        <xsl:value-of select="FIELD[@NO='4']/DATA/COMP[@NO='5']/@TEXT"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:if>
  <xsl:if test="FIELD[@NO='4']/DATA/COMP[@NO='6']/@TEXT">
    <xsl:element name="NTE">
      <xsl:attribute name="TEXT">
        <xsl:value-of select="FIELD[@NO='4']/DATA/COMP[@NO='6']/@TEXT"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:if>
  <xsl:if test="FIELD[@NO='4']/DATA/COMP[@NO='7']/@TEXT">
    <xsl:element name="NTE">
      <xsl:attribute name="TEXT">
        <xsl:value-of select="FIELD[@NO='4']/DATA/COMP[@NO='7']/@TEXT"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:if>
  <xsl:if test="FIELD[@NO='4']/DATA/COMP[@NO='8']/@TEXT">
    <xsl:element name="NTE">
      <xsl:attribute name="TEXT">
        <xsl:value-of select="FIELD[@NO='4']/DATA/COMP[@NO='8']/@TEXT"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:if>

```

Figure D-13 ASTM_IN.xsl

The default ASTM templates

```

</xsl:template>

<!--
*****
*   END OF DOCUMENT
*****
-->
</xsl:stylesheet>

```

Figure D-13 ASTM_IN.xsl**The ASTM output templates**

This section describes the XSLT that **cobas IT 3000** uses to create ASTM messages to send to the host.

The ASTM result report template

This section gives the XSLT that **cobas IT 3000** uses to create ASTM messages sending result reports to the host. This file is ASTM_OUT.xsl.

```

<?xml version="1.0" encoding="iso-8859-1"?>
<!--
*****
*   DOCUMENT ELEMENT: xsl:stylesheet
*****
-->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:output method="xml" indent="yes" version="1.0" encoding="UTF-8"/>

  <!--
  *****
  *   Parameter definitions
  *****
  -->

  <xsl:param name="transferComments">
    <!-- <data name="transferComments" comment="Shall comments be transferred?" type="boolean" />
  -->
    <xsl:value-of select="false()" />
  </xsl:param>

  <xsl:param name="test">
    <!-- <data comment="Configuration test parameter" type="boolean" /> -->
    <xsl:value-of select="false()" />
  </xsl:param>

  <!--
  *****
  *   Entry points
  *****
  -->
  <xsl:template match="/">

```

Figure D-14 ASTM_OUT.xsl

```

    <xsl:element name="ASTM">
      <xsl:apply-templates select="*" mode="header"/>
      <xsl:apply-templates select="*" mode="body"/>
      <xsl:apply-templates select="*" mode="footer"/>
    </xsl:element>
  </xsl:template>

  <xsl:template match="*" mode="header">
    <xsl:apply-templates select="*" mode="header"/>
  </xsl:template>

  <xsl:template match="*" mode="body">
    <xsl:apply-templates select="*" mode="body"/>
  </xsl:template>

  <xsl:template match="*" mode="footer">
    <xsl:apply-templates select="*" mode="footer"/>
  </xsl:template>

  <!--
  *****
  *   MSH - mode "header"
  *****
  -->
  <xsl:template match="MSH" mode="header">

    <xsl:element name="REC">
      <xsl:attribute name="ID">H</xsl:attribute>
      <xsl:attribute name="TYPE">HEADER</xsl:attribute>

      <xsl:element name="FIELD">
        <xsl:attribute name="NO">2</xsl:attribute>
        <xsl:attribute name="TEXT">|^&amp;</xsl:attribute>
      </xsl:element>

      <xsl:element name="FIELD">
        <xsl:attribute name="NO">5</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@SENDING_APPLICATION"/>
        </xsl:attribute>
      </xsl:element>

      <xsl:element name="FIELD">
        <xsl:attribute name="NO">10</xsl:attribute>
        <xsl:attribute name="TEXT">
          <xsl:value-of select="@RECEIVING_APPLICATION"/>
        </xsl:attribute>
      </xsl:element>

      <xsl:element name="FIELD">
        <xsl:attribute name="NO">12</xsl:attribute>
        <xsl:attribute name="TEXT">P</xsl:attribute>
      </xsl:element>

      <xsl:element name="FIELD">

```

Figure D-14 ASTM_OUT.xsl

```

        <xsl:attribute name="NO">14</xsl:attribute>
        <xsl:attribute name="TEXT">
            <xsl:value-of select="@DATE"/>
        </xsl:attribute>
    </xsl:element>

    <xsl:apply-templates select="*" mode="header"/>
</xsl:element>
</xsl:template>

<!--
*****
*   QRD - mode "body"
*****
-->
<xsl:template match="QRD" mode="body">

    <xsl:element name="REC">

        <xsl:attribute name="ID">Q</xsl:attribute>

        <xsl:element name="FIELD">
            <xsl:attribute name="NO">2</xsl:attribute>
            <xsl:attribute name="TEXT">1</xsl:attribute>
        </xsl:element>

        <xsl:element name="FIELD">
            <xsl:attribute name="NO">3</xsl:attribute>
            <xsl:element name="DATA">
                <xsl:element name="COMP">
                    <xsl:attribute name="NO">2</xsl:attribute>
                    <xsl:attribute name="TEXT">
                        <xsl:value-of select="@SAMPLEID"/>
                    </xsl:attribute>
                </xsl:element>
            </xsl:element>
        </xsl:element>

        <xsl:element name="FIELD">
            <xsl:attribute name="NO">5</xsl:attribute>
            <xsl:attribute name="TEXT">ALL</xsl:attribute>
        </xsl:element>

        <xsl:element name="FIELD">
            <xsl:attribute name="NO">13</xsl:attribute>
            <xsl:attribute name="TEXT">0</xsl:attribute>
        </xsl:element>

    </xsl:element>

</xsl:template>

<!--
*****
*   PID - mode "body"
*****
-->

```

Figure D-14 ASTM_OUT.xsl

```

*****
-->
<xsl:template match="PID" mode="body">

  <xsl:element name="REC">
    <xsl:attribute name="ID">P</xsl:attribute>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">1</xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="ORC/OBR/@SAMPLEID"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">6</xsl:attribute>
      <xsl:element name="DATA">
        <xsl:element name="COMP">
          <xsl:attribute name="NO">1</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@LASTNAME"/>
          </xsl:attribute>
        </xsl:element>
        <xsl:element name="COMP">
          <xsl:attribute name="NO">2</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@FIRSTNAME"/>
          </xsl:attribute>
        </xsl:element>
        <xsl:element name="COMP">
          <xsl:attribute name="NO">3</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@NAMEX"/>
          </xsl:attribute>
        </xsl:element>
        <xsl:element name="COMP">
          <xsl:attribute name="NO">5</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@TITLE"/>
          </xsl:attribute>
        </xsl:element>
      </xsl:element>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">8</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@DOB"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>

```

Figure D-14 ASTM_OUT.xsl

```

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">9</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SEX"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">11</xsl:attribute>
      <xsl:element name="DATA">
        <xsl:element name="COMP">
          <xsl:attribute name="NO">1</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@STREET"/>
          </xsl:attribute>
        </xsl:element>
        <xsl:element name="COMP">
          <xsl:attribute name="NO">2</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@CITY"/>
          </xsl:attribute>
        </xsl:element>
        <xsl:element name="COMP">
          <xsl:attribute name="NO">4</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@POSTCODE"/>
          </xsl:attribute>
        </xsl:element>
      </xsl:element>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">12</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@INSURER"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:apply-templates select="*" mode="orc"/>
  </xsl:element>
</xsl:template>

<!--
*****
*   ORC/OBR applied by PID
*****
-->
<xsl:template match="ORC/OBR" mode="orc">
  <xsl:element name="REC">
    <xsl:attribute name="ID">0</xsl:attribute>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">1</xsl:attribute>

```

Figure D-14 ASTM_OUT.xsl


```

</xsl:element>

<xsl:element name="FIELD">
  <xsl:attribute name="NO">3</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@SAMPLEID"/>
  </xsl:attribute>
</xsl:element>

<xsl:element name="FIELD">
  <xsl:attribute name="NO">4</xsl:attribute>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT"/>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="OBX/@TRAY"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="OBX/@POSITION"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">4</xsl:attribute>
      <xsl:attribute name="TEXT"/>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">5</xsl:attribute>
      <xsl:attribute name="TEXT"/>
    </xsl:element>
    <xsl:element name="COMP">
      <xsl:attribute name="NO">6</xsl:attribute>
      <xsl:attribute name="TEXT"/>
    </xsl:element>
  </xsl:element>
</xsl:element>

<xsl:element name="FIELD">
  <xsl:attribute name="NO">5</xsl:attribute>
  <xsl:attribute name="TEXT">ALL</xsl:attribute>
</xsl:element>

<xsl:element name="FIELD">
  <xsl:attribute name="NO">7</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="../@REQUESTDATE"/>
  </xsl:attribute>
</xsl:element>

```

Figure D-14 ASTM_OUT.xsl

```

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">26</xsl:attribute>
      <xsl:attribute name="TEXT">F</xsl:attribute>
    </xsl:element>

    <xsl:apply-templates select="*" mode="obx"/>
  </xsl:element>
</xsl:template>

<!--
*****
*   OBX applied by ORC/OBR
*****
-->
<xsl:template match="OBX" mode="obx">

  <xsl:element name="REC">
    <xsl:attribute name="ID">R</xsl:attribute>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SET"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:element name="DATA">
        <xsl:element name="COMP">
          <xsl:attribute name="NO">4</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@ID"/>
          </xsl:attribute>
        </xsl:element>
      </xsl:element>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">4</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@VALUE"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">5</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@UNITS"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:if test="./NTE/@TEXT != ''">
      <xsl:element name="FIELD">
        <xsl:attribute name="NO">7</xsl:attribute>

```

Figure D-14 ASTM_OUT.xsl

```

        <xsl:attribute name="TEXT">A</xsl:attribute>
      </xsl:element>
    </xsl:if>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">9</xsl:attribute>
      <xsl:attribute name="TEXT">F</xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">12</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@INSTRUMENT_MEASUREMENT_TIME"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">13</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@COMPLETEDATE"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">14</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@VALIDSTAT"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">22</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:if test="@CONFIDENTIAL">
          <xsl:choose>
            <xsl:when test="@CONFIDENTIAL=1">C</xsl:when>
            <xsl:otherwise>N</xsl:otherwise>
          </xsl:choose>
        </xsl:if>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>

  <xsl:if test="$transferComments">
    <xsl:apply-templates select="NTE" mode="nte"/>
  </xsl:if>
</xsl:template>

<!--
*****
*   NTE applied by ORC/OBR/NTE
*****
-->

```

Figure D-14 ASTM_OUT.xsl

```

<xsl:template match="NTE" mode="nte">
  <xsl:element name="REC">
    <xsl:attribute name="ID">C</xsl:attribute>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SET"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">4</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@TEXT"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:template>

<!--
*****
*   MSH - mode "footer"
*****
-->
<xsl:template match="MSH" mode="footer">
  <xsl:element name="REC">
    <xsl:attribute name="ID">L</xsl:attribute>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">1</xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">N</xsl:attribute>
    </xsl:element>

    <xsl:apply-templates select="*" mode="footer"/>
  </xsl:element>
</xsl:template>

<!--
*****
*   END OF DOCUMENT
*****
-->
</xsl:stylesheet><!-- Stylus Studio meta-information - (c) 2004-2005. Progress Software Corporation.
All rights reserved.
<metaInformation>
<scenarios/><MapperMetaTag><MapperInfo srcSchemaPathIsRelative="yes" srcSchemaInterpretAsXML="no"
destSchemaPath="" destSchemaRoot="" destSchemaPathIsRelative="yes" destSchemaInterpretAsXML="no"/
><MapperBlockPosition></MapperBlockPosition></MapperMetaTag>

```

Figure D-14 ASTM_OUT.xsl

```
</metaInformation>
-->
```

Figure D-14 ASTM_OUT.xsl

The ASTM quality control output template

This section gives the XSLT used for ASTM quality control result messages sent to the host. This file is ASTM-QualityResultsOUT.xsl.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!--
*****
*   DOCUMENT ELEMENT: xsl:stylesheet
*****
-->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >

    <xsl:output method="xml" indent="yes" version="1.0" encoding="UTF-8"/>

    <!--
    *****
    *   Parameter definitions
    *****
    -->
    <!-- <xsl:param name="test" select="false()"> -->
        <!-- <data comment="Configuration test parmeter" datatype="boolean" /> -->
    <!-- </xsl:param> -->

    <!--
    *****
    *   Entry points
    *****
    -->
    <xsl:template match="/">
        <xsl:element name="ASTM">
            <xsl:apply-templates select="*" mode="header"/>
            <xsl:apply-templates select="*" mode="body"/>
            <xsl:apply-templates select="*" mode="footer"/>
        </xsl:element>
    </xsl:template>

    <xsl:template match="*" mode="header">
        <xsl:apply-templates select="*" mode="header"/>
    </xsl:template>

    <xsl:template match="*" mode="body">
        <xsl:apply-templates select="*" mode="body"/>
    </xsl:template>

    <xsl:template match="*" mode="footer">
        <xsl:apply-templates select="*" mode="footer"/>
    </xsl:template>

    <!--
```

Figure D-15 ASTM-QualityResultsOUT.xsl

```

*****
*   MSH - mode "header"
*****
-->
<xsl:template match="MSH" mode="header">

  <xsl:element name="REC">
    <xsl:attribute name="ID">H</xsl:attribute>
    <xsl:attribute name="TYPE">HEADER</xsl:attribute>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">|\^&amp;</xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">5</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SENDING_APPLICATION"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">10</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@RECEIVING_APPLICATION"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">12</xsl:attribute>
      <xsl:attribute name="TEXT">P</xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">14</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@DATE"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:apply-templates select="*" mode="header"/>

  </xsl:element>

</xsl:template>

<!--
*****
*   PID - mode "body"
*****
-->
<xsl:template match="PID" mode="body">

  <xsl:element name="REC">

```

Figure D-15 ASTM-QualityResultsOUT.xsl

```

        <xsl:attribute name="ID">P</xsl:attribute>

        <xsl:element name="FIELD">
            <xsl:attribute name="NO">2</xsl:attribute>
            <xsl:attribute name="TEXT">1</xsl:attribute>
            <xsl:if test="$test" >
                <xsl:attribute name="TEST">
                    <xsl:value-of select="$test"/>
                </xsl:attribute>
            </xsl:if>
        </xsl:element>
        <xsl:apply-templates select="*" mode="orc"/>

    </xsl:element>

</xsl:template>

<!--
*****
*   ORC/OBR applied by PID
*****
-->
<xsl:template match="ORC/OBR" mode="orc">
    <xsl:element name="REC">
        <xsl:attribute name="ID">O</xsl:attribute>

        <xsl:element name="FIELD">
            <xsl:attribute name="NO">2</xsl:attribute>
            <xsl:attribute name="TEXT">1</xsl:attribute>
        </xsl:element>

        <xsl:element name="FIELD">

            <xsl:attribute name="NO">3</xsl:attribute>
            <xsl:attribute name="TEXT">
<value-of select="@CONTROLID" />
<!--<xsl:variable name="ctrlID" select="@CONTROLID"/>
                <xsl:choose>
<xsl:when test="$ctrlID='40'">MyControl</xsl:when>
<xsl:otherwise><value-of select="@CONTROLID" /></xsl:otherwise>
                </xsl:choose> -->
            </xsl:attribute>
        </xsl:element>

        <xsl:element name="FIELD">
            <xsl:attribute name="NO">26</xsl:attribute>
            <xsl:attribute name="TEXT">F</xsl:attribute>
        </xsl:element>

        <xsl:apply-templates select="*" mode="obx"/>

    </xsl:element>

</xsl:template>

```

Figure D-15 ASTM-QualityResultsOUT.xsl

```

<!--
*****
*   OBX applied by ORC/OBR
*****
-->
<xsl:template match="OBX" mode="obx">

  <xsl:element name="REC">
    <xsl:attribute name="ID">R</xsl:attribute>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SET"/>
      </xsl:attribute>
    </xsl:element>

    <xsl:element name="FIELD">
      <xsl:attribute name="NO">3</xsl:attribute>
    </xsl:element>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">4</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ID"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
  <xsl:element name="DATA">
    <xsl:element name="COMP">
      <xsl:attribute name="NO">5</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@ANALYTNR"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">4</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@VALUE"/>
    </xsl:attribute>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">5</xsl:attribute>
    <xsl:attribute name="TEXT">
      <xsl:value-of select="@UNITS"/>
    </xsl:attribute>
  </xsl:element>
  <xsl:element name="FIELD">
    <xsl:attribute name="NO">7</xsl:attribute>
    <xsl:attribute name="TEXT">N</xsl:attribute>
  </xsl:element>

```

Figure D-15 ASTM-QualityResultsOUT.xsl


```

        </xsl:element>

        <xsl:element name="FIELD">
        <xsl:attribute name="NO">12</xsl:attribute>
        <xsl:attribute name="TEXT">
            <xsl:value-of select="@INSTRUMENT_MEASUREMENT_TIME"/>
        </xsl:attribute>
        </xsl:element>

        <xsl:element name="FIELD">
            <xsl:attribute name="NO">13</xsl:attribute>
            <xsl:attribute name="TEXT">
                <xsl:value-of select="@COMPLETEDATE"/>
            </xsl:attribute>
        </xsl:element>

    <xsl:element name="FIELD">
        <xsl:attribute name="NO">14</xsl:attribute>
        <xsl:attribute name="TEXT">
            <xsl:value-of select="../@INSTRUMENT"/>
        </xsl:attribute>
    </xsl:element>

</xsl:element>
</xsl:template>

<!--
*****
*   MSH - mode "footer"
*****
-->
<xsl:template match="MSH" mode="footer">
    <xsl:element name="REC">
        <xsl:attribute name="ID">L</xsl:attribute>

        <xsl:element name="FIELD">
            <xsl:attribute name="NO">2</xsl:attribute>
            <xsl:attribute name="TEXT">1</xsl:attribute>
        </xsl:element>

        <xsl:element name="FIELD">
            <xsl:attribute name="NO">3</xsl:attribute>
            <xsl:attribute name="TEXT">N</xsl:attribute>
        </xsl:element>

        <xsl:apply-templates select="*" mode="footer"/>
    </xsl:element>
</xsl:template>

<!--
*****
*   END OF DOCUMENT
*****
-->
</xsl:stylesheet>

```

Figure D-15 ASTM-QualityResultsOUT.xsl

The ASTM sample event templates

This section gives the XSLT that **cobas IT 3000** uses to create the ASTM outgoing sample event messages. This file is SampleSeen_Astm.xml.

```
<?xml version="1.0" encoding="windows-1252"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" indent="yes" version="1.0" encoding="UTF-8"/>

  <xsl:param name="ASTM_20">
    <!-- <data name="ASTM_20" comment="Should patient and order records be sent" type="boolean" /> -->
  </xsl:param>

  <xsl:param name="SENDING_APPLICATION">
    <!-- <data name="SENDING_APPLICATION" comment="The name of the sending application"
type="string" /> -->
    <xsl:value-of select="'" />
  </xsl:param>

  <xsl:param name="SENDING_FACILITY">
    <!-- <data name="SENDING_FACILITY" comment="The name of the sending facility" type="string" /> -->
  </xsl:param>

  <xsl:param name="RECEIVING_APPLICATION">
    <!-- <data name="RECEIVING_APPLICATION" comment="The name of the receiving application"
type="string" /> -->
    <xsl:value-of select="'" />
  </xsl:param>

  <xsl:param name="RECEIVING_FACILITY">
    <!-- <data name="RECEIVING_FACILITY" comment="The name of the receiving facility" type="string"
/> -->
    <xsl:value-of select="'" />
  </xsl:param>

  <xsl:param name="SEND_ONCE">
    <!-- <data name="SEND_ONCE" type="string" comment="List of events for sending messages only once"
/> -->
    <xsl:value-of select="'" />
  </xsl:param>

  <xsl:param name="SEND_INSTRUMENTS">
    <!-- <data name="INSTRUMENTS" type="string" comment="List of instruments" /> -->
    <xsl:value-of select="ALL" />
  </xsl:param>

  <xsl:param name="SEND_LOCATIONS">
    <!-- <data name="Locations" type="string" comment="List of locations" /> -->
    <xsl:value-of select="ALL" />
  </xsl:param>

  <xsl:param name="L_SENDING_APPLICATION">
```

Figure D-16 SampleSeen_Astm.xml

```

<xsl:choose>
  <xsl:when test="string-length($SENDING_APPLICATION) != 0">
    <xsl:value-of select="$SENDING_APPLICATION"/>
  </xsl:when>
  <xsl:otherwise>
    <xsl:value-of select="/SampleEvent/MSH/@SENDING_APPLICATION"/>
  </xsl:otherwise>
</xsl:choose>
</xsl:param>
<xsl:param name="L_SENDING_FACILITY">
  <xsl:choose>
    <xsl:when test="string-length($SENDING_FACILITY) != 0">
      <xsl:value-of select="$SENDING_FACILITY"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="/SampleEvent/MSH/@SENDING_FACILITY"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:param>
<xsl:param name="L_RECEIVING_APPLICATION">
  <xsl:choose>
    <xsl:when test="string-length($RECEIVING_APPLICATION) != 0">
      <xsl:value-of select="$RECEIVING_APPLICATION"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="/SampleEvent/MSH/@RECEIVING_APPLICATION"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:param>
<xsl:param name="L_RECEIVING_FACILITY">
  <xsl:choose>
    <xsl:when test="string-length($RECEIVING_FACILITY) != 0">
      <xsl:value-of select="$RECEIVING_FACILITY"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select="/SampleEvent/MSH/@RECEIVING_FACILITY"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:param>

<xsl:template match="/">
  <xsl:element name="ASTM">
    <xsl:apply-templates select="/SampleEvent/MSH" mode="header"/>
    <xsl:if test="$ASTM_20">
      <xsl:apply-templates select="/SampleEvent/EQU/PID" mode="patient"/>
      <xsl:apply-templates select="/SampleEvent/EQU/PID/SAC" mode="order"/>
    </xsl:if>
    <xsl:apply-templates select="/SampleEvent/EQU" mode="M1"/>
    <xsl:apply-templates select="/SampleEvent/EQU/PID/SAC" mode="M2"/>
    <xsl:apply-templates select="/" mode="terminator"/>
  </xsl:element>
</xsl:template>
<xsl:template match="MSH" mode="header">
  <xsl:element name="REC">
    <xsl:attribute name="ID">H</xsl:attribute>

```

Figure D-16 SampleSeen_Astm.xsl

```

<xsl:attribute name="TYPE">HEADER</xsl:attribute>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">2</xsl:attribute>
  <xsl:attribute name="TEXT">|^&amp;</xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">5</xsl:attribute>
  <DATA>
    <COMP>
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_SENDING_APPLICATION"/>
      </xsl:attribute>
    </COMP>
    <COMP>
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_SENDING_FACILITY"/>
      </xsl:attribute>
    </COMP>
    <COMP>
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_RECEIVING_APPLICATION"/>
      </xsl:attribute>
    </COMP>
    <COMP>
      <xsl:attribute name="NO">4</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="$L_RECEIVING_FACILITY"/>
      </xsl:attribute>
    </COMP>
  </DATA>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">12</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@PROCESSING_ID"/>
  </xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">14</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@EVENT_DATE_TIME"/>
  </xsl:attribute>
</xsl:element>
</xsl:template>
<xsl:template match="EQU" mode="M1">
  <xsl:element name="REC">
    <xsl:attribute name="ID">M</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">1</xsl:attribute>
    </xsl:element>
  </xsl:element>

```

Figure D-16 SampleSeen_Astm.xsl

```

<xsl:element name="FIELD">
  <xsl:attribute name="NO">3</xsl:attribute>
  <xsl:attribute name="TEXT">EQU</xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">4</xsl:attribute>
  <DATA>
    <COMP>
      <xsl:attribute name="NO">4</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@EQUIPMENT_IDENTIFIER_NAME"/>
      </xsl:attribute>
    </COMP>
    <COMP>
      <xsl:attribute name="NO">5</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@EQUIPMENT_IDENTIFIER_CLIENT"/>
      </xsl:attribute>
    </COMP>
    <COMP>
      <xsl:attribute name="NO">6</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@EQUIPMENT_IDENTIFIER_MODULE"/>
      </xsl:attribute>
    </COMP>
    <COMP>
      <xsl:attribute name="NO">7</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@EQUIPMENT_IDENTIFIER_EVENT"/>
      </xsl:attribute>
    </COMP>
  </DATA>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">5</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@EVENT_DATE_TIME"/>
  </xsl:attribute>
</xsl:element>
</xsl:template>
<xsl:template match="SAC" mode="M2">
  <xsl:element name="REC">
    <xsl:attribute name="ID">M</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">2</xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">SAC</xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">6</xsl:attribute>
      <xsl:attribute name="TEXT">

```

Figure D-16 SampleSeen_Astm.xsl

```

    <xsl:value-of select="@CONTAINER_IDENTIFIER"/>
  </xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">7</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@PRIMARY_CONTAINER_IDENTIFIER"/>
  </xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">9</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@SPECIMEN_SOURCE"/>
  </xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">10</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@REGISTRATION_DATE_TIME"/>
  </xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">11</xsl:attribute>
  <DATA>
    <COMP>
      <xsl:attribute name="NO">1</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@CONTAINER_STATUS"/>
      </xsl:attribute>
    </COMP>
    <COMP>
      <xsl:attribute name="NO">4</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SPECIFIC_CONTAINER_STATUS"/>
      </xsl:attribute>
    </COMP>
  </DATA>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">12</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@CARRIER_TYPE"/>
  </xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">13</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@CARRIER_IDENTIFIER"/>
  </xsl:attribute>
</xsl:element>
<xsl:element name="FIELD">
  <xsl:attribute name="NO">14</xsl:attribute>
  <xsl:attribute name="TEXT">
    <xsl:value-of select="@POSITION_IN_CARRIER"/>
  </xsl:attribute>

```

Figure D-16 SampleSeen_Astm.xsl

```

    </xsl:element>
  </xsl:element>
</xsl:template>
<xsl:template match="MSH" mode="terminator">
  <xsl:element name="REC">
    <xsl:attribute name="ID">L</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">1</xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">N</xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:template>
<xsl:template match="PID" mode="patient">
  <xsl:element name="REC">
    <xsl:attribute name="ID">P</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">1</xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@EXTERNAL_PATIENT_ID"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">4</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@PATIENT_ID"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">6</xsl:attribute>
      <DATA>
        <COMP>
          <xsl:attribute name="NO">1</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@LAST_NAME"/>
          </xsl:attribute>
        </COMP>
        <COMP>
          <xsl:attribute name="NO">2</xsl:attribute>
          <xsl:attribute name="TEXT">
            <xsl:value-of select="@FIRST_NAME"/>
          </xsl:attribute>
        </COMP>
      </DATA>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">8</xsl:attribute>
      <xsl:attribute name="TEXT">

```

Figure D-16 SampleSeen_Astm.xsl

```

        <xsl:value-of select="@DATE_OF_BIRTH"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">9</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SEX"/>
      </xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:template>
<xsl:template match="SAC" mode="order">
  <xsl:element name="REC">
    <xsl:attribute name="ID">0</xsl:attribute>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">2</xsl:attribute>
      <xsl:attribute name="TEXT">1</xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">3</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@CONTAINER_IDENTIFIER"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">8</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="../../PID/@ORDER_DATE"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">12</xsl:attribute>
      <xsl:attribute name="TEXT">X</xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">15</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="../../PID/@ORDER_DATE"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">16</xsl:attribute>
      <xsl:attribute name="TEXT">
        <xsl:value-of select="@SPECIMEN_SOURCE"/>
      </xsl:attribute>
    </xsl:element>
    <xsl:element name="FIELD">
      <xsl:attribute name="NO">26</xsl:attribute>
      <xsl:attribute name="TEXT">I</xsl:attribute>
    </xsl:element>
  </xsl:element>
</xsl:template>
</xsl:stylesheet>

```

Figure D-16 SampleSeen_Astm.xsl

XML schema definitions

The definition of the XML used in host communications

This chapter contains the XML Schema Definitions (XSD) of the XML files that **cobas IT 3000** uses in host communications. To configure the way that **cobas IT 3000** processes host messages, you can edit the XSLT files that convert the files that comply with these XSD formats.

In this chapter	Chapter 12
XML schema definitions	D-119
XML schema definition of HL7 and ASTM message structure	D-120
XML schema definition of messages used by cobas IT 3000	D-122

XML schema definitions

This chapter contains the schema definitions of the XML files that **cobas IT 3000** uses with ASTM and HL7 messages.

The **cobas IT 3000** uses an XML format to describe the structure of HL7 or ASTM messages.

👁 For the definition of the format to describe the structure of HL7 or ASTM messages, see *XML schema definition of HL7 and ASTM message structure* on page D-120.

The **cobas IT 3000** uses an XML format to describe the logical content of messages. You need to convert the data in messages into this format before **cobas IT 3000** can write the data to its internal database. Similarly, when **cobas IT 3000** retrieves data from its internal database, it is initially in this format.

👁 For the definition of the format that describes the logical content of messages, see *XML schema definition of messages used by cobas IT 3000* on page D-122.

There are XSLT stylesheets that convert in both directions between these two formats. You can configure how **cobas IT 3000** handles HL7 and ASTM messages by editing these XSLT stylesheets.

XML schema definition of HL7 and ASTM message structure

This section contains the XML Schema Definition (XSD) for the XML files that describe the structure of ASTM or HL7 messages.

When **cobas IT 3000** reads in an ASTM or HL7 message, it automatically transforms it into an XML document that follows this XSD. To send an ASTM or HL7 message, **cobas IT 3000** must read data formatted according to this XSD.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" version="2.03.11">

  <!-- Message type looks the same, but element distinguishes protocol -->
  <xs:element name="ASTM" type="messageType"/>
  <xs:element name="HL7" type="messageType"/>

  <xs:complexType name="messageType">
    <xs:sequence>
      <xs:element name="REC" type="recordType" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attributeGroup ref="messageAttr"/>
  </xs:complexType>

  <xs:attributeGroup name="messageAttr">
    <xs:attribute name="DIRECTION" type="xs:string"/>
    <xs:attribute name="MODULE" type="xs:string"/>
    <xs:attribute name="SOURCE" type="xs:string"/>
    <xs:attribute name="DESTINATION" type="xs:string"/>
    <xs:attribute name="KEY" type="xs:string"/>
  </xs:attributeGroup>

  <xs:complexType name="recordType">
    <xs:sequence>
      <xs:element name="FIELD" type="fieldType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="REC" type="recordType" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attributeGroup ref="recordAttr"/>
  </xs:complexType>

  <xs:attributeGroup name="recordAttr">
    <xs:attribute name="ID" type="xs:string"/>
    <xs:attribute name="TYPE" type="xs:string" use="optional"/>
  </xs:attributeGroup>

  <xs:complexType name="fieldType">
    <xs:sequence>
      <xs:element name="DATA" type="dataType" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attributeGroup ref="fieldAttr"/>
  </xs:complexType>

  <xs:attributeGroup name="fieldAttr">
    <xs:attribute name="NO" type="xs:integer"/>
    <xs:attribute name="TEXT" type="xs:string"/>
  </xs:attributeGroup>
</xs:schema>
```

Figure D-17 XSD of the XML that defines an ASTM or HL7 message

```
<xs:complexType name="dataType">
  <xs:sequence>
    <xs:element name="COMP" type="componentType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attributeGroup ref="fieldAttr"/>
</xs:complexType>

<xs:complexType name="componentType">
  <xs:sequence>
    <xs:element name="SUB" type="subComponentType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attributeGroup ref="fieldAttr"/>
</xs:complexType>

<xs:complexType name="subComponentType">
  <xs:attributeGroup ref="fieldAttr"/>
</xs:complexType>

</xs:schema>
```

Figure D-17 XSD of the XML that defines an ASTM or HL7 message

XML schema definition of messages used by cobas IT 3000

This section contains the XML Schema definition of the format that **cobas IT 3000** requires data to be in.

The **cobas IT 3000** must have the data in this format, in order to save the data to its internal database, and to update its user interface. Also, when **cobas IT 3000** first retrieves data from its internal database, it is in this format.

You cannot send orders containing results

Although the default templates and the XSD contain fields for orders containing results, this is not supported in the current release. This may be supported in future releases. If you require this functionality, please contact Roche Diagnostics Global Customer support.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" version="2.03.11">

  <!--Messages-->
  <xs:element name="QueryMessage" type="queryMessageType"/>

  <xs:element name="OrderResults" type="orderResultsType"/>

  <xs:element name="OrderResponse" type="orderResponseType"/>

  <xs:element name="QualityResults" type="orderResultsType"/>

  <xs:element name="SEE" type="sampleSeenType"/>

  <xs:element name="OrderRequest" type="orderRequestType"/>

  <xs:element name="PatientMessage" type="patientMessageType"/>

  <!--Message Types-->

  <xs:complexType name="queryMessageType">
    <xs:sequence>
      <xs:element name="MSH" minOccurs="1" maxOccurs="1" type="mshTypeOut"/>
      <xs:element name="QRD" minOccurs="1" maxOccurs="1" type="qrdType"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="orderResultsType">
    <xs:sequence>
      <xs:element name="MSH" minOccurs="1" maxOccurs="1" type="mshTypeOut"/>
      <xs:element name="PID" minOccurs="1" maxOccurs="1" type="pidTypeOut"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="orderResponseType">
    <xs:sequence>
      <xs:element name="MSH" minOccurs="1" maxOccurs="1" type="mshTypeOut"/>
      <xs:element name="MSA" minOccurs="1" maxOccurs="1" type="msaType"/>
    </xs:sequence>
  </xs:complexType>

</xs:schema>
```

Figure D-18 XSD of the XML that **cobas IT 3000** uses to define the structure of a host message

```

</xs:complexType>

<xs:complexType name="sampleSeenType">
  <xs:sequence>
    <xs:element name="MSH" minOccurs="1" maxOccurs="1" type="mshTypeOut"/>
    <xs:element name="EQU" minOccurs="1" maxOccurs="1" type="equType"/>
    <xs:element name="LOC" minOccurs="1" maxOccurs="1" type="locType"/>
    <xs:element name="SAMPLE" minOccurs="1" maxOccurs="1" type="sampleType"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="orderRequestType">
  <xs:sequence>
    <xs:element name="MSH" minOccurs="1" maxOccurs="1" type="mshTypeIn"/>
    <xs:element name="PID" minOccurs="1" maxOccurs="1" type="pidTypeIn"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="patientMessageType">
  <xs:sequence>
    <xs:element name="MSH" minOccurs="1" maxOccurs="1" type="mshTypeIn"/>
    <xs:element name="PID" minOccurs="1" maxOccurs="1" type="pidTypeIn"/>
  </xs:sequence>
</xs:complexType>

<!--complex types-->
<xs:complexType name="mshTypeIn">
  <xs:attribute name="SENDER" type="xs:string"/>
  <xs:attributeGroup ref="mshAttrInOut"/>
</xs:complexType>

<xs:complexType name="mshTypeOut">
  <xs:attributeGroup ref="mshAttrInOut"/>
  <xs:attributeGroup ref="mshAttrOut"/>
</xs:complexType>

<xs:complexType name="msaType">
  <xs:attribute name="SEQ" type="xs:string"/>
  <xs:attribute name="ACK" type="xs:string"/>
  <xs:attribute name="TEXT" type="xs:string"/>
</xs:complexType>

<xs:attributeGroup name="mshAttrInOut">
  <xs:attribute name="ACK_TYPE" type="xs:string"/>
  <xs:attribute name="ACCEPT_TYPE" type="xs:string"/>
  <xs:attribute name="CONTROL_ID" type="xs:string"/>
  <xs:attribute name="MESSAGE_TYPE" type="xs:string"/>
  <xs:attribute name="EVENT_TYPE" type="xs:string"/>
</xs:attributeGroup>

<xs:attributeGroup name="mshAttrOut">
  <xs:attribute name="FORMAT" type="xs:string"/>
  <xs:attribute name="SENDING_APPLICATION" type="xs:string"/>
  <xs:attribute name="SENDING_FACILITY" type="xs:string"/>
  <xs:attribute name="RECEIVING_APPLICATION" type="xs:string"/>

```

Figure D-18 XSD of the XML that **cobas IT 3000** uses to define the structure of a host message

```

<xs:attribute name="RECEIVING_FACILITY" type="xs:string"/>
<xs:attribute name="DATE" type="xs:string"/>
<xs:attribute name="PROCESSING_ID" type="xs:string"/>
<xs:attribute name="VERSION" type="xs:string"/>
<xs:attribute name="ENCODING" type="xs:string"/>
</xs:attributeGroup>

<xs:complexType name="pidTypeIn">
  <xs:sequence minOccurs="1" maxOccurs="unbounded">
    <xs:choice>
      <xs:element name="PD1" type="pd1TypeIn"/>
      <xs:element name="NK1" type="nk1TypeIn"/>
      <xs:element name="PV1" type="pv1TypeIn"/>
      <xs:element name="ORC" type="orcTypeIn"/>
    </xs:choice>
  </xs:sequence>
  <xs:attributeGroup ref="pidAttrInOut"/>
  <xs:attributeGroup ref="pidAttrIn"/>
</xs:complexType>

<xs:complexType name="pd1TypeIn">
  <xs:attribute name="PD16" type="xs:string"/>
  <xs:attribute name="PD17" type="xs:string"/>
  <xs:attribute name="PD18" type="xs:string"/>
  <xs:attribute name="PD19" type="xs:string"/>
  <xs:attribute name="PD20" type="xs:string"/>
</xs:complexType>

<xs:complexType name="nk1TypeIn">
  <xs:attribute name="NK1" type="xs:string"/>
  <xs:attribute name="NK2" type="xs:string"/>
  <xs:attribute name="NK3" type="xs:string"/>
  <xs:attribute name="NK4" type="xs:string"/>
  <xs:attribute name="NK5" type="xs:string"/>
  <xs:attribute name="NK6" type="xs:string"/>
  <xs:attribute name="NK7" type="xs:string"/>
  <xs:attribute name="NK8" type="xs:string"/>
  <xs:attribute name="NK9" type="xs:string"/>
  <xs:attribute name="NK10" type="xs:string"/>
  <xs:attribute name="NK11" type="xs:string"/>
  <xs:attribute name="NK12" type="xs:string"/>
  <xs:attribute name="NK13" type="xs:string"/>
  <xs:attribute name="NK14" type="xs:string"/>
  <xs:attribute name="NK15" type="xs:string"/>
  <xs:attribute name="NK16" type="xs:string"/>
  <xs:attribute name="NK17" type="xs:string"/>
  <xs:attribute name="NK18" type="xs:string"/>
  <xs:attribute name="NK19" type="xs:string"/>
  <xs:attribute name="NK20" type="xs:string"/>
</xs:complexType>

<xs:complexType name="pidTypeOut">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:element name="PV1" type="pv1TypeOut" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="ORC" type="orcTypeOut"/>
  </xs:sequence>
</xs:complexType>

```

Figure D-18 XSD of the XML that **cobas IT 3000** uses to define the structure of a host message


```

</xs:sequence>
<xs:attributeGroup ref="pidAttrInOut"/>
<xs:attributeGroup ref="pidAttrOut"/>
</xs:complexType>

<xs:attributeGroup name="pidAttrInOut">
  <xs:attribute name="PATID" type="xs:string"/>
  <xs:attribute name="LASTNAME" type="xs:string"/>
  <xs:attribute name="FIRSTNAME" type="xs:string"/>
  <xs:attribute name="DOB" type="xs:string"/>
  <xs:attribute name="SEX" type="xs:string"/>
  <xs:attribute name="MORE1" type="xs:string"/>
  <xs:attribute name="MORE2" type="xs:string"/>
  <xs:attribute name="MORE3" type="xs:string"/>
  <xs:attribute name="MORE4" type="xs:string"/>
  <xs:attribute name="MORE5" type="xs:string"/>
  <xs:attribute name="MORE6" type="xs:string"/>
  <xs:attribute name="MORE7" type="xs:string"/>
  <xs:attribute name="MORE8" type="xs:string"/>
  <xs:attribute name="MORE9" type="xs:string"/>
  <xs:attribute name="MORE10" type="xs:string"/>
  <xs:attribute name="MORE11" type="xs:string"/>
  <xs:attribute name="MORE12" type="xs:string"/>
  <xs:attribute name="MORE13" type="xs:string"/>
  <xs:attribute name="MORE14" type="xs:string"/>
  <xs:attribute name="MORE15" type="xs:string"/>
</xs:attributeGroup>

<xs:attributeGroup name="pidAttrIn">
  <xs:attribute name="DOBFMT" type="xs:string"/>
  <xs:attribute name="DATEFMT" type="xs:string"/>
  <xs:attribute name="FLAGS" type="xs:string"/>
  <xs:attribute name="HIS" type="xs:string"/>
  <xs:attribute name="NAMEX" type="xs:string"/>
  <xs:attribute name="STREET" type="xs:string"/>
  <xs:attribute name="POSTCODE" type="xs:string"/>
  <xs:attribute name="CITY" type="xs:string"/>
  <xs:attribute name="INSURER" type="xs:string"/>
  <xs:attribute name="TITLE" type="xs:string"/>
  <xs:attribute name="ADMITDATE" type="xs:string"/>
  <xs:attribute name="DISCHARGEDATE" type="xs:string"/>
  <xs:attribute name="PD1" type="xs:string"/>
  <xs:attribute name="PD2" type="xs:string"/>
  <xs:attribute name="PD3" type="xs:string"/>
  <xs:attribute name="PD4" type="xs:string"/>
  <xs:attribute name="PD5" type="xs:string"/>
  <xs:attribute name="PD6" type="xs:string"/>
  <xs:attribute name="PD7" type="xs:string"/>
  <xs:attribute name="PD8" type="xs:string"/>
  <xs:attribute name="PD9" type="xs:string"/>
  <xs:attribute name="PD10" type="xs:string"/>
  <xs:attribute name="PD11" type="xs:string"/>
  <xs:attribute name="PD12" type="xs:string"/>
  <xs:attribute name="PD13" type="xs:string"/>
  <xs:attribute name="PD14" type="xs:string"/>

```

Figure D-18 XSD of the XML that **cobas IT 3000** uses to define the structure of a host message

```

    <xs:attribute name="PD15" type="xs:string"/>
  </xs:attributeGroup>

  <xs:attributeGroup name="pidAttrOut">
    <xs:attribute name="SET" type="xs:string"/>
  </xs:attributeGroup>

  <xs:complexType name="pv1TypeIn">
    <xs:sequence>
      <xs:element name="PV2" type="pv2Type"/>
      <xs:element name="DG1" type="dglType" minOccurs="1" maxOccurs="unbounded"/>
      <xs:element name="IN1" type="in1Type" minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attributeGroup ref="pv1AttrInOut"/>
    <xs:attributeGroup ref="pv1AttrIn"/>
  </xs:complexType>

  <xs:complexType name="pv1TypeOut">
    <xs:attributeGroup ref="pv1AttrInOut"/>
    <xs:attributeGroup ref="pv1AttrOut"/>
  </xs:complexType>

  <xs:attributeGroup name="pv1AttrIn">
    <xs:attribute name="WARD" type="xs:string"/>
    <xs:attribute name="SPECIALITY" type="xs:string"/>
    <xs:attribute name="ROOM" type="xs:string"/>
  </xs:attributeGroup>

  <xs:attributeGroup name="pv1AttrOut">
    <xs:attribute name="SET" type="xs:string"/>
    <xs:attribute name="ADMITDATE" type="xs:string"/>
    <xs:attribute name="DISCHARGEDATE" type="xs:string"/>
  </xs:attributeGroup>

  <xs:attributeGroup name="pv1AttrInOut">
    <xs:attribute name="VISITNR" type="xs:string"/>
    <xs:attribute name="PV1" type="xs:string"/>
    <xs:attribute name="PV2" type="xs:string"/>
    <xs:attribute name="PV3" type="xs:string"/>
    <xs:attribute name="PV4" type="xs:string"/>
    <xs:attribute name="PV5" type="xs:string"/>
    <xs:attribute name="PV6" type="xs:string"/>
    <xs:attribute name="PV7" type="xs:string"/>
    <xs:attribute name="PV8" type="xs:string"/>
    <xs:attribute name="PV9" type="xs:string"/>
    <xs:attribute name="PV10" type="xs:string"/>
    <xs:attribute name="PV11" type="xs:string"/>
    <xs:attribute name="PV12" type="xs:string"/>
    <xs:attribute name="PV13" type="xs:string"/>
    <xs:attribute name="PV14" type="xs:string"/>
    <xs:attribute name="PV15" type="xs:string"/>
  </xs:attributeGroup>

  <xs:complexType name="pv2Type">
    <xs:attribute name="PV16" type="xs:string"/>
  </xs:complexType>

```

Figure D-18 XSD of the XML that **cobas IT 3000** uses to define the structure of a host message

```

<xs:attribute name="PV17" type="xs:string"/>
<xs:attribute name="PV18" type="xs:string"/>
<xs:attribute name="PV19" type="xs:string"/>
<xs:attribute name="PV20" type="xs:string"/>
</xs:complexType>

<xs:complexType name="dg1Type">
  <xs:attribute name="ICDCODE" type="xs:string"/>
  <xs:attribute name="TEXT" type="xs:string"/>
  <xs:attribute name="TYPE" type="xs:string"/>
  <xs:attribute name="DATE" type="xs:string"/>
  <xs:attribute name="DATEFMT" type="xs:string"/>
  <xs:attribute name="DG1" type="xs:string"/>
  <xs:attribute name="DG2" type="xs:string"/>
  <xs:attribute name="DG3" type="xs:string"/>
  <xs:attribute name="DG4" type="xs:string"/>
  <xs:attribute name="DG5" type="xs:string"/>
  <xs:attribute name="DG6" type="xs:string"/>
  <xs:attribute name="DG7" type="xs:string"/>
  <xs:attribute name="DG8" type="xs:string"/>
  <xs:attribute name="DG9" type="xs:string"/>
  <xs:attribute name="DG10" type="xs:string"/>
  <xs:attribute name="DG11" type="xs:string"/>
  <xs:attribute name="DG12" type="xs:string"/>
  <xs:attribute name="DG13" type="xs:string"/>
  <xs:attribute name="DG14" type="xs:string"/>
  <xs:attribute name="DG15" type="xs:string"/>
  <xs:attribute name="DG16" type="xs:string"/>
  <xs:attribute name="DG17" type="xs:string"/>
  <xs:attribute name="DG18" type="xs:string"/>
  <xs:attribute name="DG19" type="xs:string"/>
  <xs:attribute name="DG20" type="xs:string"/>
</xs:complexType>

<xs:complexType name="in1Type">
  <xs:sequence>
    <xs:element name="IN2" type="in2Type"/>
  </xs:sequence>
  <xs:attribute name="IN1" type="xs:string"/>
  <xs:attribute name="IN2" type="xs:string"/>
  <xs:attribute name="IN3" type="xs:string"/>
  <xs:attribute name="IN4" type="xs:string"/>
  <xs:attribute name="IN5" type="xs:string"/>
  <xs:attribute name="IN6" type="xs:string"/>
  <xs:attribute name="IN7" type="xs:string"/>
  <xs:attribute name="IN8" type="xs:string"/>
  <xs:attribute name="IN9" type="xs:string"/>
  <xs:attribute name="IN10" type="xs:string"/>
  <xs:attribute name="IN11" type="xs:string"/>
  <xs:attribute name="IN12" type="xs:string"/>
  <xs:attribute name="IN13" type="xs:string"/>
  <xs:attribute name="IN14" type="xs:string"/>
  <xs:attribute name="IN15" type="xs:string"/>
</xs:complexType>

```

Figure D-18 XSD of the XML that **cobas IT 3000** uses to define the structure of a host message

```

<xs:complexType name="in2Type">
  <xs:attribute name="IN16" type="xs:string"/>
  <xs:attribute name="IN17" type="xs:string"/>
  <xs:attribute name="IN18" type="xs:string"/>
  <xs:attribute name="IN19" type="xs:string"/>
  <xs:attribute name="IN20" type="xs:string"/>
</xs:complexType>

<xs:complexType name="orcTypeIn">
  <xs:sequence>
    <xs:element name="ORDERER" type="ordererTypeIn" minOccurs="0" maxOccurs="1"/>
    <xs:element name="NTE" type="nteTypeIn" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="OBR" type="obrTypeIn" minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attributeGroup ref="orcAttrInOut"/>
  <xs:attributeGroup ref="orcAttrIn"/>
</xs:complexType>

<xs:complexType name="orcTypeOut">
  <xs:sequence>
    <xs:element name="NTE" type="nteTypeOut" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="OBR" type="obrTypeOut" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attributeGroup ref="orcAttrInOut"/>
  <xs:attributeGroup ref="orcAttrOut"/>
</xs:complexType>

<xs:attributeGroup name="orcAttrInOut">
  <xs:attribute name="LOCATION" type="xs:string"/>
  <xs:attribute name="SAMPLEID" type="xs:string"/>
  <xs:attribute name="ORIGINAL_SAMPLEID" type="xs:string"/>
  <xs:attribute name="SPECIMEN" type="xs:string"/>
  <xs:attribute name="ORDERNR" type="xs:string"/>
  <xs:attribute name="HOST_ORDER_ID" type="xs:string"/>
  <xs:attribute name="PRIORITY" type="xs:string"/>
  <xs:attribute name="ORDERER" type="xs:string"/>
  <xs:attribute name="WITHDRAWDATE" type="xs:string"/>
  <xs:attribute name="PHYSICIAN_CODE" type="xs:string"/>
  <xs:attribute name="PHYSICIAN_NAME" type="xs:string"/>
</xs:attributeGroup>

<xs:attributeGroup name="orcAttrIn">
  <xs:attribute name="DESTINATION" type="xs:string"/>
  <xs:attribute name="CONTROL" type="xs:string"/>
  <xs:attribute name="STATUS" type="xs:string"/>
  <xs:attribute name="SPECIALITY" type="xs:string"/>
  <xs:attribute name="ROOM" type="xs:string"/>
  <xs:attribute name="PHYSICIAN_CREATE" type="xs:string"/>
  <xs:attribute name="DIAGNOSIS" type="xs:string"/>
  <xs:attribute name="MEDICATION" type="xs:string"/>
  <xs:attribute name="PREGNANCY" type="xs:string"/>
  <xs:attribute name="MENSTRUAL" type="xs:string"/>
  <xs:attribute name="HOSTDATA" type="xs:string"/>
  <xs:attribute name="INSURANCE" type="xs:string"/>
  <xs:attribute name="ORDERDATE" type="xs:string"/>

```

Figure D-18 XSD of the XML that **cobas IT 3000** uses to define the structure of a host message

```

    <xs:attribute name="ORDERDATEFMT" type="xs:string"/>
    <xs:attribute name="WITHDRAWDATEFMT" type="xs:string"/>
</xs:attributeGroup>

<xs:attributeGroup name="orcAttrOut">
    <xs:attribute name="SET" type="xs:string"/>
    <xs:attribute name="CONTROLID" type="xs:string"/>
    <xs:attribute name="REQUESTDATE" type="xs:string"/>
    <xs:attribute name="REPORTING_AVENUE" type="xs:string"/>
    <xs:attribute name="HOSTDATA" type="xs:string"/>
    <xs:attribute name="FULL_ORDERNR" type="xs:string"/>
    <xs:attribute name="CONTROL_NAME" type="xs:string"/>
    <xs:attribute name="CONTROL_LOT" type="xs:string"/>
    <xs:attribute name="GERNAME" type="xs:string"/>
</xs:attributeGroup>

<xs:complexType name="ordererTypeIn">
    <xs:attributeGroup ref="ordererAttrIn"/>
</xs:complexType>

<xs:attributeGroup name="ordererAttrIn">
    <xs:attribute name="DEFAULT_AUFGCD" type="xs:string"/>
    <xs:attribute name="HOSTCODE" use="required">
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:maxLength value="5"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="NAME" type="xs:string" use="required"/>
    <xs:attribute name="REMARK" type="xs:string"/>
    <xs:attribute name="COSTCENTER" type="xs:string"/>
    <xs:attribute name="SORT" type="xs:integer"/>
    <xs:attribute name="TITLE1" type="xs:string"/>
    <xs:attribute name="TITLE2" type="xs:string"/>
    <xs:attribute name="STREET" type="xs:string"/>
    <xs:attribute name="ZIP" type="xs:string"/>
    <xs:attribute name="CITY" type="xs:string"/>
    <xs:attribute name="BILLINGCODE" type="xs:string"/>
    <!-- WARD is pseudo-boolean: Must = "J" else results not sent -->
    <xs:attribute name="WARD" type="xs:string"/>
    <xs:attribute name="TOUR" type="xs:string"/>
    <xs:attribute name="ORDERERGROUP" type="xs:string"/>
    <xs:attribute name="ORGANIZATION" type="xs:string"/>
    <xs:attribute name="ABBREVIATION" type="xs:string"/>
    <xs:attribute name="PHONE" type="xs:string"/>
    <xs:attribute name="FAX" type="xs:string"/>
    <xs:attribute name="WARDID" type="xs:string" use="required"/>
</xs:attributeGroup>

<xs:complexType name="nteTypeIn">
    <xs:attributeGroup ref="nteAttrInOut"/>
    <xs:attributeGroup ref="nteAttrIn"/>
</xs:complexType>

```

Figure D-18 XSD of the XML that **cobas IT 3000** uses to define the structure of a host message

```

<xs:complexType name="nteTypeOut">
  <xs:attributeGroup ref="nteAttrInOut"/>
  <xs:attributeGroup ref="nteAttrOut"/>
</xs:complexType>

<xs:attributeGroup name="nteAttrIn">
  <xs:attribute name="TEXTTYPE" type="xs:string"/>
</xs:attributeGroup>

<xs:attributeGroup name="nteAttrInOut">
  <xs:attribute name="TEXTCODE" type="xs:string"/>
  <xs:attribute name="TEXT" type="xs:string"/>
  <xs:attribute name="INTERNAL" type="xs:string"/>
</xs:attributeGroup>

<xs:attributeGroup name="nteAttrOut">
  <xs:attribute name="SET" type="xs:string"/>
  <xs:attribute name="SOURCE" type="xs:string"/>
</xs:attributeGroup>

<xs:complexType name="obrTypeIn">
  <xs:sequence>
    <xs:element name="OBX" type="obxTypeIn" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attributeGroup ref="obrAttrInOut"/>
  <xs:attributeGroup ref="obrAttrIn"/>
</xs:complexType>

<xs:complexType name="obrTypeOut">
  <xs:sequence>
    <xs:element name="NTE" type="nteTypeOut" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="OBX" type="obxTypeOut" minOccurs="1" maxOccurs="unbounded"/>
    <xs:element name="QRD" type="qrdType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attributeGroup ref="obrAttrInOut"/>
  <xs:attributeGroup ref="obrAttrOut"/>
</xs:complexType>

<xs:complexType name="qrdType">
  <xs:attribute name="SET" type="xs:string"/>
  <xs:attribute name="SAMPLEID" type="xs:string"/>
  <xs:attribute name="TRAY" type="xs:string"/>
  <xs:attribute name="INSTRUMENT" type="xs:string"/>
  <xs:attribute name="SPECIMEN" type="xs:string"/>
  <xs:attribute name="POSITION" type="xs:string"/>
  <xs:attribute name="WHAT" type="xs:string"/>
</xs:complexType>

<xs:attributeGroup name="obrAttrIn">
  <xs:attribute name="TESTCODE" type="xs:string"/>
  <xs:attribute name="SPECIMEN" type="xs:string"/>
  <xs:attribute name="SECIMEN" type="xs:string"/>
  <xs:annotation>
    <xs:documentation>Warning misspelled SPECIMEN in code</xs:documentation>
  </xs:annotation>

```

Figure D-18 XSD of the XML that **cobas IT 3000** uses to define the structure of a host message

```

</xs:attribute>
<xs:attribute name="ACTION" type="xs:string"/>
<xs:attribute name="PRIORITY" type="xs:string"/>
<xs:attribute name="DILUTION" type="xs:string"/>
</xs:attributeGroup>

<xs:attributeGroup name="obrAttrOut">
  <xs:attribute name="SET" type="xs:string"/>
  <xs:attribute name="SPECIMEN" type="xs:string"/>
  <xs:attribute name="ORDERNR" type="xs:string"/>
  <xs:attribute name="GROUP" type="xs:string"/>
  <xs:attribute name="PROFILE" type="xs:string"/>
  <xs:attribute name="PROFILENAME" type="xs:string"/>
  <xs:attribute name="ID" type="xs:string"/>
</xs:attributeGroup>

<xs:attributeGroup name="obrAttrInOut">
  <xs:attribute name="SAMPLEID" type="xs:string"/>
  <xs:attribute name="INSTRUMENT" type="xs:string"/>
</xs:attributeGroup>

<xs:complexType name="obxTypeIn">
  <xs:sequence>
    <xs:element name="NTE" type="nteobxTypeIn" minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attributeGroup ref="obxAttrInOut"/>
</xs:complexType>

<xs:complexType name="obxTypeOut">
  <xs:sequence>
    <xs:element name="NTE" type="nteobxTypeOut" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attributeGroup ref="obxAttrInOut"/>
  <xs:attributeGroup ref="obxAttrOut"/>
</xs:complexType>

<xs:complexType name="nteobxTypeIn">
  <xs:attributeGroup ref="nteobxAttrInOut"/>
  <xs:attributeGroup ref="nteobxAttrIn"/>
</xs:complexType>

<xs:complexType name="nteobxTypeOut">
  <xs:attributeGroup ref="nteobxAttrInOut"/>
</xs:complexType>

<xs:attributeGroup name="nteobxAttrInOut">
  <xs:attribute name="TEXT" type="xs:string"/>
  <xs:attribute name="SET" type="xs:string"/>
  <xs:attribute name="CODE" type="xs:string"/>
  <xs:attribute name="DESC" type="xs:string"/>
  <xs:attribute name="SOURCE" type="xs:string"/>
</xs:attributeGroup>

<xs:attributeGroup name="nteobxAttrIn">
  <xs:attribute name="TEXTCODE" type="xs:string"/>

```

Figure D-18 XSD of the XML that **cobas IT 3000** uses to define the structure of a host message

```

    <xs:attribute name="TEXTTYPE" type="xs:string"/>
    <xs:attribute name="INTERNAL" type="xs:string"/>
  </xs:attributeGroup>

  <xs:attributeGroup name="obxAttrInOut">
    <xs:attribute name="SET" type="xs:string"/>
    <xs:attribute name="ID" type="xs:string"/>
    <xs:attribute name="ANALYTNR" type="xs:string"/>
    <xs:attribute name="VALUE" type="xs:string"/>
    <xs:attribute name="UNITS" type="xs:string"/>
    <xs:attribute name="COMPLETEDATE" type="xs:string"/>
    <xs:attribute name="INDICATOR" type="xs:string"/>
    <xs:attribute name="INDIKATOR" type="xs:string"/>
    <xs:attribute name="INSTRUMENT_MEASUREMENT_TIME" type="xs:string"/>
    <xs:attribute name="VALIDSTAT" type="xs:string"/>
    <xs:attribute name="GERNR" type="xs:string"/>
    <xs:attribute name="GERNAME" type="xs:string"/>
    <xs:attribute name="MEDVALUID" type="xs:string"/>
    <xs:attribute name="MEDVALDATE" type="xs:string"/>
    <xs:attribute name="RELEASEUID" type="xs:string"/>
    <xs:attribute name="RELEASEDATE" type="xs:string"/>
    <xs:attribute name="REFBEREICH" type="xs:string"/>
    <xs:attribute name="PROFILE" type="xs:string"/>
    <xs:attribute name="ONORMAL" type="xs:string"/>
    <xs:attribute name="UNORMAL" type="xs:string"/>
  </xs:attributeGroup>

  <xs:attributeGroup name="obxAttrOut">
    <xs:attribute name="SAMPLEID" type="xs:string"/>
    <xs:attribute name="VERDFAKT" type="xs:string"/>
    <xs:attribute name="AUTOMVERDFAKT" type="xs:string"/>
    <xs:attribute name="TRAY" type="xs:string"/>
    <xs:attribute name="POSITION" type="xs:string"/>
    <xs:attribute name="SPECIMEN" type="xs:string"/>
    <xs:attribute name="DECIMAL_PLACES" type="xs:string"/>
    <xs:attribute name="GRAPH" type="xs:string"/>
    <xs:attribute name="LOCATION" type="xs:string"/>
    <xs:attribute name="DAYNUMBER" type="xs:string"/>
    <xs:attribute name="SYNONYM" type="xs:string"/>
    <xs:attribute name="NAME" type="xs:string"/>
    <xs:attribute name="1" type="xs:string"/>
    <xs:attribute name="2" type="xs:string"/>
    <xs:attribute name="3" type="xs:string"/>
    <xs:attribute name="HOSTCODE" type="xs:string"/>
    <xs:attribute name="STATISTIC" type="xs:string"/>
    <xs:attribute name="LEISTUNG" type="xs:string"/>
    <xs:attribute name="PRICE" type="xs:string"/>
    <xs:attribute name="POINTS" type="xs:string"/>
    <xs:attribute name="COMMENT" type="xs:string"/>
    <xs:attribute name="NUMALPHAFLAG" type="xs:string"/>
    <xs:attribute name="VALIDSTATALL" type="xs:string"/>
    <xs:attribute name="RESULTCOLOR" type="xs:string"/>
    <xs:attribute name="CONFIDENTIAL" type="xs:string"/>
  </xs:attributeGroup>

```

Figure D-18 XSD of the XML that **cobas IT 3000** uses to define the structure of a host message


```
<xs:complexType name="equType">
  <xs:attribute name="ACTION" type="xs:string"/>
  <xs:attribute name="TYPE" type="xs:string"/>
  <xs:attribute name="DATETIME" type="xs:string"/>
  <xs:attribute name="ID" type="xs:string"/>
</xs:complexType>

<xs:complexType name="locType">
  <xs:attribute name="LOCATION" type="xs:string"/>
  <xs:attribute name="CONTAINER" type="xs:string"/>
  <xs:attribute name="POSITION" type="xs:string"/>
</xs:complexType>

<xs:complexType name="sampleType">
  <xs:attribute name="SPECIMEN" type="xs:string"/>
  <xs:attribute name="ID" type="xs:string"/>
</xs:complexType>
</xs:schema>
```

Figure D-18 XSD of the XML that **cobas IT 3000** uses to define the structure of a host message

