



MediPort Communicator

User Manual for Service Providers and Hospitals

Document version 1.13.0

Doc. recipient:	external
Print date:	23 March 2012
Author:	MediData AG
Copyright:	© 2012 MediData AG, Root Längenbold



Document management

Document storage:

Format: Microsoft Word 2007

Documentation amendment history

Rev.	Reason for amendment	Date	Initials
1.0	First version	06.10.2004	sum
1.1	Revision for MPCommunicator Version 1.2.0	19.08.2003	sum
1.2	Revision for MPCommunicator Version 1.3.0	05.04.2004	sum
1.3	Revision for MPCommunicator Version 1.4.0	28.09.2004	sum
1.4	Revision and reorganisation of chapters	31.10.2004	sum
1.5	Revision for MPCommunicator Version 1.5.0	28.04.2005	sum
1.6	Revision for MPCommunicator Version 1.6.0	01.07.2005	sum
1.6.1	XML Schema GetDocumentList changed	12.08.2005	DbA
1.6.2	Multi client supports now up to 100 clients	28.09.2005	scr
1.7	Revision for MPCommunicator Version 1.7.0	21.11.2005	sca
1.8	Revision for MPCommunicator Version 1.8.0	23.01.2006	sca
1.9	Revision for MPCommunicator Version 1.9.0	10.08.2007	sca
1.9.1	Compatibility for Windows Vista added	22.08.2008	sca
1.9.2	Revision for MPCommunicator Version 1.9.2	12.08.2010	sca
1.9.3	Revision for MPCommunicator Version 1.9.3	17.01.2011	sca
1.10.0	Revision for MPCommunicator Version 1.10.0	24.03.2011	sca
1.11.0	Revision for MPCommunicator Version 1.11.0	29.11.2011	sca
1.12.0	Revision for MPCommunicator Version 1.12.0	30.01.2012	sca
1.13.0	Revision for MPCommunicator Version 1.13.0	23.03.2012	sca



Table of contents

1	INTRODUCTION.....	6
1.1	What is MediPort?	6
1.2	General	7
1.3	Positioning	8
1.4	TP and TG workflow	9
1.4.1	“Tiers Payant” (TP) workflow	9
1.4.2	“Tiers Garant” (TG or TGM) workflow.....	11
2	BASIC PRINCIPLES.....	13
2.1	Objectives.....	13
2.2	Architecture.....	13
2.3	System requirements	14
2.3.1	Supported operating systems	14
2.3.2	Hardware requirements	14
2.3.3	Software requirements	15
3	FUNCTIONS	16
3.1	Sending data	16
3.1.1	Application flow	16
3.1.2	Send control file	17
3.2	Receiving data	22
3.2.1	General	22
3.2.2	Multi User.....	22
3.2.3	XSLT Transformation	22
3.2.4	eKARUS to XML 4.3 conversion	23
3.3	Address book.....	24
3.3.1	General	24
3.3.2	Address book set up	24
3.4	Status tracking.....	25
3.4.1	Function description	25
3.4.2	Return file	25
3.4.3	XML schema.....	26
3.5	Sending without send control files.....	28
3.5.1	Functional overview	28
3.5.2	Configuration of the parameters	28
3.5.3	Configuration of the task.....	30
3.6	Archiving	32
3.7	Generic Tasks	34
3.7.1	Functional overview	34
3.7.2	Implementing generic tasks.....	34
3.7.3	Configuration of generic tasks	34
3.8	Multi-user functionality	36
3.8.1	Description of function	36
3.9	Security functions / certificates	37

MediData

3.9.1	Authentication and encryption	37
3.9.2	Certificate handling	37
3.10	Process control.....	38
3.10.1	Timer / Task Manager.....	38
3.10.2	External trigger	38
3.11	Logging.....	39
3.11.1	Log message layout.....	39
4	INSTALLATION	40
4.1	General	40
4.1.1	Windows 7 / Windows Vista requirements	40
4.1.2	Initial installation	40
4.1.3	Directory structure after installation	41
4.2	Windows specific.....	42
4.2.1	Installation as Windows service (Optional).....	42
4.3	Linux / Unix specific	43
4.3.1	Linux / Unix Demon start script mpc.sh	43
4.3.2	Installation as Linux / Unix Demon (Optional)	43
5	CONFIGURATION	44
5.1	Minimal configuration	44
5.2	Optional configuration	45
5.2.1	Interval scheduler	45
5.2.2	Checks before sending.....	46
5.2.3	Send control.....	46
5.2.4	Trigger process control (RMI)	48
5.2.5	Proxy configuration	49
5.2.6	Path configuration.....	50
5.2.7	Document status.....	53
5.3	Advanced configuration	54
5.3.1	Cron scheduler	54
5.3.2	Multiple MPC services	55
5.3.3	SMB File Server Access.....	55
5.4	Logger configuration	57
5.4.1	General	57
5.4.2	Mail (SMTP).....	57
5.4.3	SNMP trap sender	57
5.5	Configuration test.....	59
5.5.1	Conditions	59
5.5.2	Execution	59
6	UPDATE	60
6.1	Update.....	60
7	OPERATION	61
7.1	General	61
7.1.1	Start-up	61

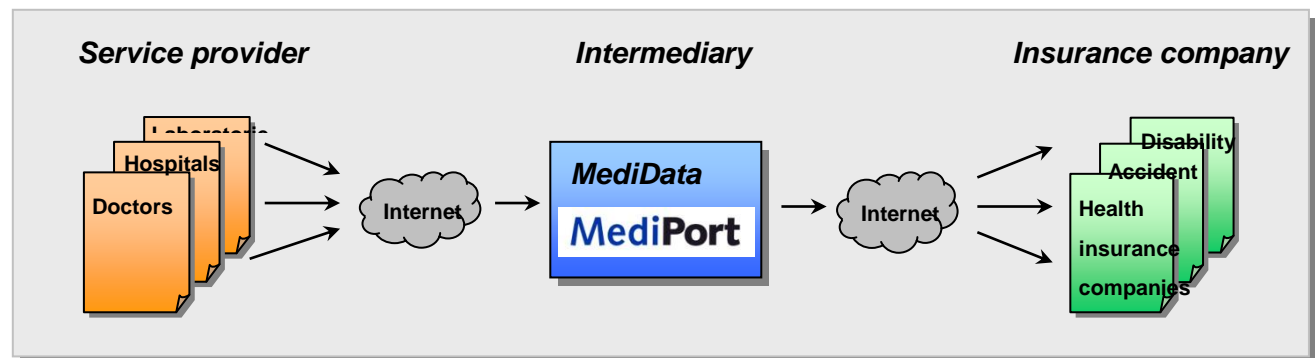
7.1.2	Operation	61
7.1.3	Stopping.....	61
7.2	Windows specific.....	63
7.2.1	Start-up	63
7.2.2	Stopping.....	63
8	MAINTENANCE.....	65
8.1	Change of certificates	65
8.2	Application	65
8.3	Database.....	65
9	UNINSTALLATION.....	66
9.1	General	66
10	TROUBLESHOOTING.....	67
10.1	Problem with Microsoft proxy	67
11	APPENDIX A: FILE NAMES OF RECEIVED DOCUMENTS	68
12	APPENDIX B: LOG MESSAGES	72
12.1	INFO (I0000 – I9999) log level messages.....	72
12.2	WARN (W0000 – W9999) log level messages	73
12.3	ERROR (E0000 – E9999) log level messages	75
12.4	FATAL (F0000 – F9999) log level messages	80
13	APPENDIX C: CONFIGURATION AND LOGGING	83
13.1	MPC configuration files	83
13.1.1	Configuration file	83
13.1.2	Windows service configuration file	89
13.2	MP Communicator logging.....	91
13.2.1	Log configuration file.....	92
13.2.2	MPC log file	93
14	APPENDIX D: SCHEDULER CONFIGURATION.....	96
14.1	Cron Expressions.....	96
15	APPENDIX E: RELEASE NOTES	98

1 Introduction

1.1 What is MediPort?

MediPort is an Internet-based platform for the health care service, which allows service providers to generate invoices and settle accounts with insurance companies electronically. The following parties are involved in the process:

- Service provider (Sender)
- Intermediary (Facilitator)
- Insurance company(Recipient)

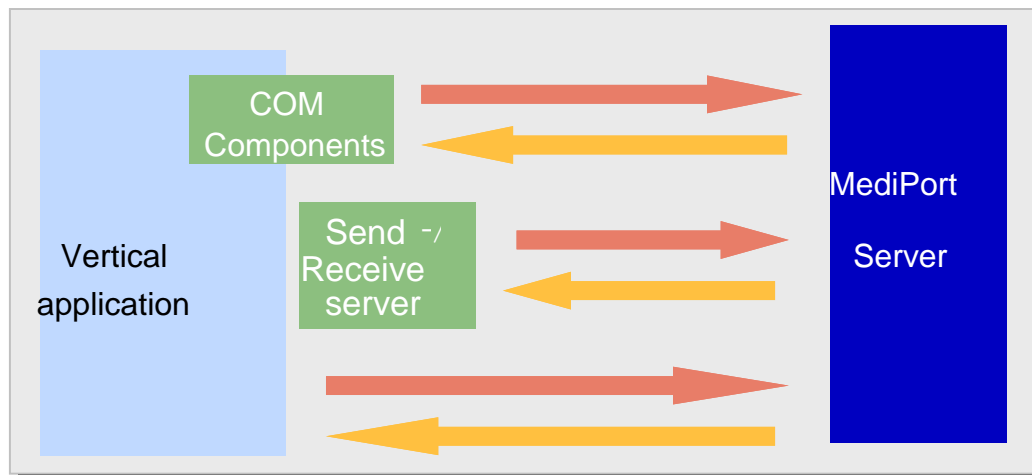


The electronic billing process shown in the above diagram between the service providers and the insurance companies applies both to electronically generated invoices and to invoice responses sent in the opposite direction.

1.2 General

The concept for connecting partners (service providers/insurance companies) to MediPort currently offers three options:

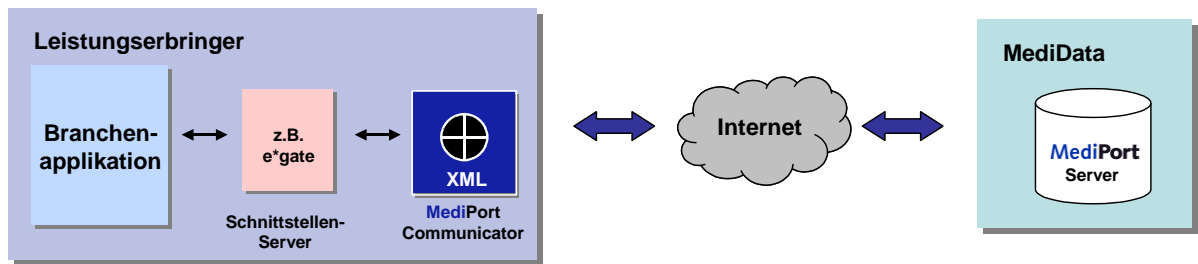
- COM components (DLL's) for integration into the vertical applications of service providers (SPs)
- MediPort Gateway, send/receive server designed for insurance companies (ICs)
- Direct connection to the MediPort Server



It has become apparent that the COM components solution is not the optimum solution for all potential service providers. There have been a number of requests for a stand-alone send/receive server particularly from hospitals. However, service providers that do not use Windows as their operating system cannot use the COM components solution and an alternative solution is required.

This solution is a send/receive server for service providers called MediPort Communicator.

1.3 Positioning



Leistungserbringer = service provider

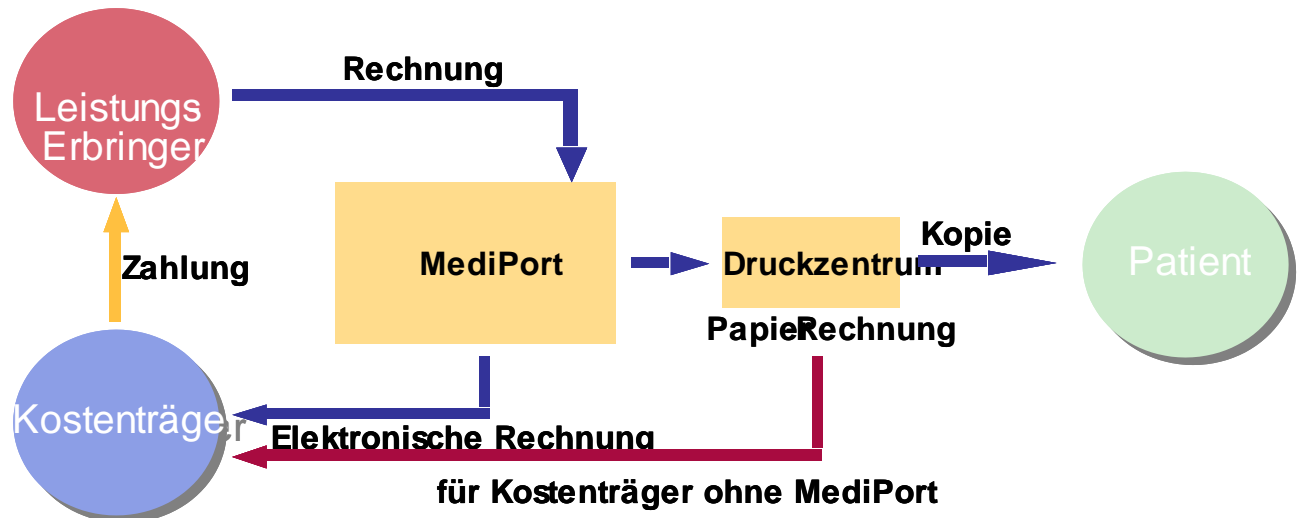
Branchenapplikation = vertical application

z.B. = e.g.

Schnittstellenserver = interface server

1.4 TP and TG workflow

1.4.1 “Tiers Payant” (TP) workflow



[Key]

Leistungserbringer = service provider

Zahlung = payment

Rechnung = invoice

Druckzentrum = print centre

Papier-Rechnung = hard copy invoice

Kopie = copy

Patient = patient

Elektronische Rechnung für Kostenträger ohne MediPort = Electronic invoice for insurance companies without MediPort

Kostenträger = Insurance company

The Tiers Payant workflow supports the delivery of documents and reminders in the Tiers Payant system. The system supports the direct transmission of documents between service providers and insurers. The following two scenarios can be distinguished depending on the respective recipient.

Recipient is part of the MediPort System

Documents are placed immediately in the recipient's outbox and can be picked up by the recipient next time a collection procedure is activated. Delivery time is minimal and the process cannot be reversed.

Recipient is not part of the MediPort System

Documents cannot be delivered electronically. MediPort hands documents over for inclusion in the printing process and documents are sent to the respective recipient (usually an insurance company) through the post.

Patients should receive a reference copy

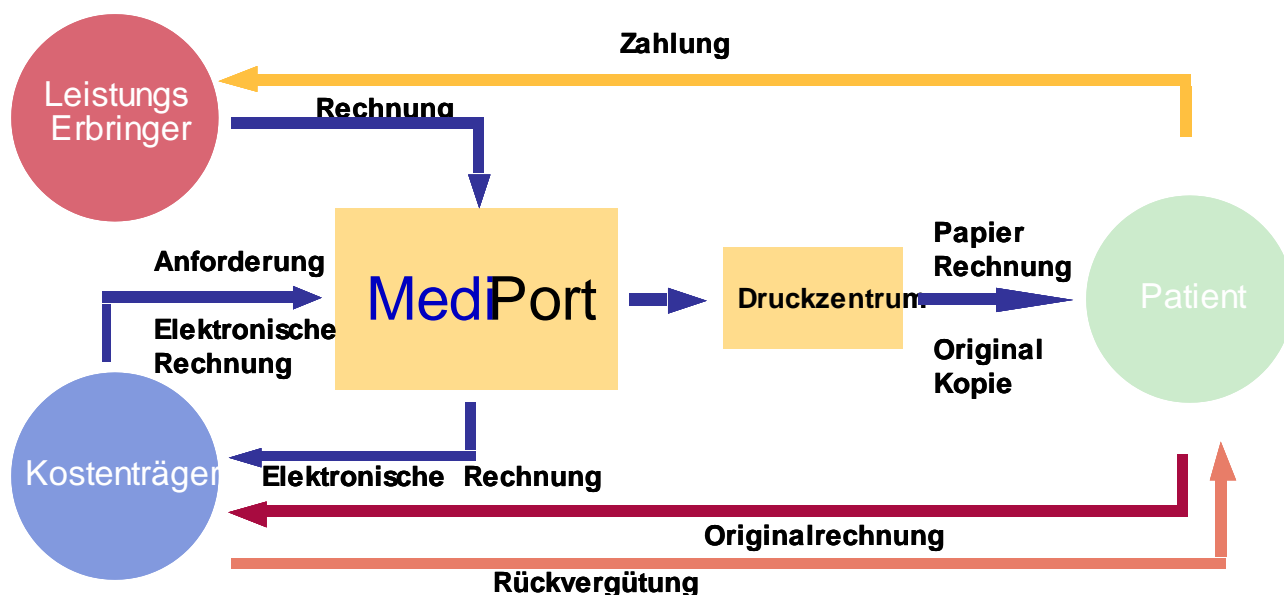
The sender can chose each time a TP document is sent whether or not a reference copy should be sent to the respective patient.

A recipient can decide when logging on or at a later point, whether or not a reference copy of each TP document should be generally and automatically printed out and sent to the respective patient.

1.4.2 “Tiers Garant” (TG or TGM) workflow

The Tiers Garant workflow supports the delivery of documents and reminders in the Tiers Garant system. The system supports the delivery of documents with the help of a printing service. In the ‘Tiers Garant’ system a distinction is made between manual and automatic versions.

Tiers Garant



[Key]

Leistungserbringer = service provider
 Zahlung = payment
 Rechnung = invoice
 Anforderung = requirement
 Elektronische Rechnung = electronic invoice
 Druckzentrum = print centre
 Papier-Rechnung = hard copy invoice
 Original Kopie = original copy
 Patient = patient
 Kostenträger = insurance company
 Original Rechnung = original invoice
 Rückvergütung = reimbursement

In the case of the manual version, documents are usually printed out and sent to the patient's address, which must appear in full in the XML document. If the patient sends the document to his insurance company, the latter can collect the data record from MediPort using the printed bar code. If the patient does not send the document within 90 days, the data record will be deleted from the MediPort server and will not be retrievable. If the sender does not request a

print out and dispatch, it will be assumed that the sender will print out the document and enter the MediPort document number on the document so that the respective insurance company can call up the data record.

TG documents are 'invisible' to recipients. They are only visible to recipients on specification of the recipient's address and the document identification number (bar code). This means that the respective recipient must be in possession of a hard copy of the document.

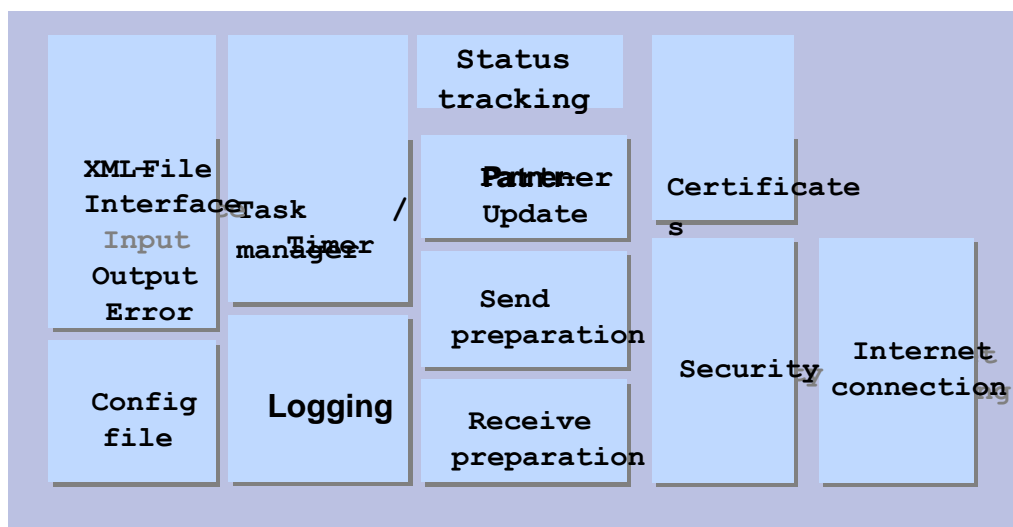
2 Basic principles

2.1 Objectives

- Basic connection to MediPort for service providers
- Separate send/receive<-> processing
- Encapsulation of security features
- Adjustment to network architecture
- Handling of all delivery types
- Universal use

2.2 Architecture

The MediPort Communicator architecture is as follows:



The product is only suitable for operation in secure networks (behind a firewall).

2.3 System requirements

2.3.1 Supported operating systems

MediPort Communicator is a pure Java application. It can be installed on various operating systems:

- Microsoft Windows from Windows 2000 up to and including Windows 7
- Mac OS X
- Unix-Derivate
 - Solaris x86 OS 5.10 and above
 - Linux

Others on request

2.3.2 Hardware requirements

Installation requires:

- 100 MB of hard disk memory for programmes (incl. Virtual Machine for Java)
- 100 MB of memory for temporary file storage (this is a guideline value and is dependent on the amount of data that needs to be stored. (Rule of thumb: 1 Tarmed XML invoice: 6-10kB; therefore approximately 10,000 invoices can be stored temporarily with 100MB)
- Minimum of 256 MB RAM (512 Mbyte recommended)
- Java Virtual Machine 1.6 (supplied as standard)

An Internet connection is essential for MediPort Communicator operation. The bandwidth required depends on the number of documents to be sent. With 256kB/s and a document size of 10 kB approximately 6,000 standard documents (256*3600/150kBit) can be sent through the system per hour. If 64kB/s are reserved for the MediPort application then approximately 1,500 standard documents can be transferred per hour. The time required for client and server authentication and the influence of server capacity utilisation is not taken into account in these calculations. Each time a new connection is established the SSL parameters are renegotiated which requires a certain amount of time.

Please note:

The bandwidth of the connection is not the only restrictive factor in Internet operations. The response times of the domain name server, the service provider's server and packet switching should also be taken into account. These times are subject to constant change and are difficult to take into account.

MediPort is extremely scaleable in terms of its architecture and the products used and adapts to the respective requirements.

2.3.3 Software requirements

MediPort Communicator requires Java Virtual Machine 1.6 (included in current release).

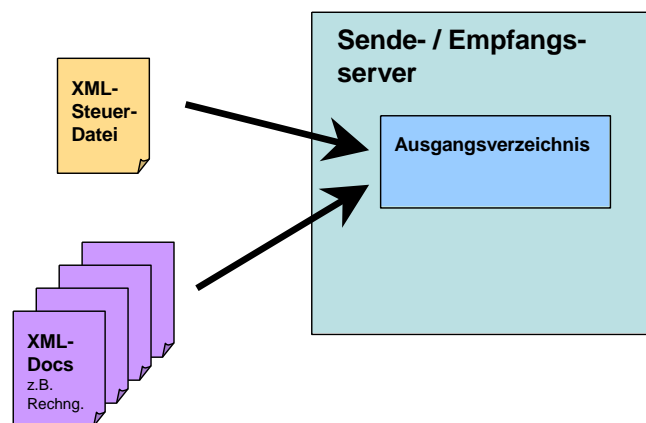
3 Functions

3.1 Sending data

3.1.1 Application flow

- The files in an output directory are sent to the recipient via MediPort.
- The **recipient's EAN** must appear in the XML file header. If the recipient's EAN number is not listed in the partner address book (recipient is not electronically connected to MediPort) the document will be printed automatically.
- XML files can be sent to www.forum-datenaustausch.ch
- Both Tiers Payant and Tiers Garant invoices can be sent.
- Before sending, the xmls are validated, the intermediary is verified and sending identical xmls multiple times is prevented.
- Files that have been sent successfully will be deleted.
- Faulty documents are recorded in an error directory and deleted from the send directory.
- Sent/received files are recorded in a log file

In order to transmit documents, MediPort Communicator requires additional information that is not visible in the XML file. This should be transferred to the MediPort Communicator with the help of a send control file, which is copied into the output directory. It is also possible to place several send control files into the output directory which are then handled consecutively.



[Key]

XML Steuerdatei = XML control file

z.B. Rechnung = e.g. invoice

Sende/Empfangserver = send/receive server

Ausgangsverzeichnis = output directory

The send process can be triggered as follows:

- The Communicator runs as an application (all operating systems), service (Windows) or Daemon [sic](Unix/Linux) and the built-in scheduler automatically triggers the send process. The scheduler can be configured as a basic interval scheduler or a cron scheduler.
- The Communicator runs as an RMI server (all operating systems) and the send process is triggered externally.

When processing an xml file it is first schema-validated according to the xsd specified in its schemaLocation attribute. The supported schemas are in the <INSTALLDIR>/xsd/ folder. Only the first 5 validation-errors of any xml are written in the log (chapter 3.11). Thereafter, a check whether an identical document has already been sent within the last hour is done. If however during this time more than 5000 documents have been sent, then only the newest 5000 documents are compared to the current one. If an xml is larger than 200 KB, then this check also does not consider the complete xml anymore, rather only the SenderDoc ID given in the send control file is compared. Finally, for all documents except those of the type XML4.3 (see below), the system checks that the MediData EAN 7601001304307 is set correctly as intermediary. In any error-case the xml is not sent but moved to an error folder instead (many of the above-mentioned parameters may be changed per configuration, and the checks may be disabled).

Peculiarities of XML4.3 documents:

- empty generalContainer_430.xsd documents cannot be sent
- if a generalContainer_430.xsd document contains an invoice (generalInvoiceRequest_430.xsd or generalInvoiceResponse_430.xsd), the recipient is determined from the invoice (regardless of any MCD), but if only an MCD is present, the MCD's recipient is taken
- encrypted or signed documents are not supported specifically
- a missing MediData EAN intermediary only generates a WARN log-entry, the send process is continued

3.1.2 Send control file

The send control file is an XML file that may contain information for 1 –n files to be sent. A record is kept in the file in respect of each file to be sent containing the following information:

- SenderDoc ID
- Filename of file to which info relates
- DocAttr
- DocPrinted
- DistType

- Subject
- DocSize
- PrintLanguage

Additional optional information:

- TrustCenterEAN
- IsPaperInvoice
- SchemaID

The send control file will be deleted once it has been processed successfully, if all files mentioned therein have been sent successfully. If xmls have been moved to the error directory, then the corresponding send control file is also moved to that directory (whereby it is renamed with a suffix *.n.xml* (*n*=integer number) to avoid overwriting any existing send control file).

The name of the send control file will be specified in the configuration file.

Default Name:

SendControl10.xml¹

Example of a SendControl10.xml file containing 2 documents

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<DocumentsToSend xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="./data/send/SendControl10.xsd">
  <Document
    SenderDocId="Your one-off invoice number"
    FileName="abcdef.xml"
    DocAttr="Tiers_Payant"
    DocPrinted="true"
    DistType="0"
    Subject="Betrifftvermerk"
    DocSize="123"
    PrintLanguage="D"/>
  <Document
    SenderDocId="Ref.723478a/34"
    FileName="xmit_23efdf.xml"
    DocAttr="Tiers_Payant"
    DocPrinted="true"
    DistType="1"
    Subject="CSS invoice"
    DocSize="4564"
    PrintLanguage="F"
    TrustCenterEAN="7601001370111"
    IsPaperInvoice="false"
    SchemaID="26020"/>
</DocumentsToSend>
```

¹ It is possible to save the records of individual files to be sent in a 'tmp' file and as soon as the file creation batch process has been completed the file name can be changed in SendControl_10.xml and the send process triggered.

</DocumentsToSend>

XML definition (Schema)

The SendControl10.xsd file containing the instructions must always be in the send directory and should not be deleted!

These instructions are specified in the schema.

SenderDocId

Any unique description relating to the sender for the corresponding document.

FileName

Name of the file to be sent. File names should contain up to 32 characters (comply with special character regulations in accordance with operating system) and must end with the character string 'xml'.

DocAttr

Defines the workflow in MediPort and must contain one the following values:

- Tiers_Payant
- Tiers_Garant_Manuell
- Tiers_Garant_Direct (only for TG DocumentResponse, PullStandard)
- direct

DocPrinted

Effect depends on the workflow:

- Tiers Garant: indicates whether or not the invoice including the reclaim voucher should be printed (exceptions may apply)
- Tiers Payant: indicates whether or not the patient's copy should be printed
- direct: irrelevant

The attribute must contain one of the following values:

"true" = no print out

"false" = print invoice/reclaim voucher/patient's copy according to workflow

DistType

Indicates the dispatch type of the printed document.

0 = B-Post (Default)

1 = A-Post

Subject

Subject line without umlauts

DocSize

Figure >= 0 indicates the size of the file

PrintLanguage

Indicates the language in which the document will be printed where appropriate. Possible enumeration values 'D', 'F', 'I'.

TrustCenterEAN

EAN number of the trust center. If the number is valid, a copy of the document will additionally be sent to that trust center. – Error -104 if EAN number is not valid.

IsPaperInvoice

Indicates whether the document is a scanned paper invoice.

Possible values: "true", "false"

SchemaID

SchemaID used by MediPort, which specifies the document type of an xmit document more precisely. This parameter is only used for documents of the type xmitdocuments_410.xsd, for any other documents it is ignored.

SendControl XML schema file (SendControl10.xsd)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns="http://www.medidata.ch/mpc/XSD" xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.medidata.ch/mpc/XSD" elementFormDefault="qualified">
  <xs:complexType name="DocumentType">
    <xs:attribute name="SenderDocId" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string"/>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="FileName" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:pattern value=".{1,256}[.][x][m][l]"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="DocAttr" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:NMTOKEN">
          <xs:enumeration value="Tiers_Payant"/>
          <xs:enumeration value="Tiers_Garant_Manuell"/>
          <xs:enumeration value="Tiers_Garant_Direct"/>
          <xs:enumeration value="direct"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="DocPrinted" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:boolean">

```

```

        <xs:pattern value="true|false"/>
    </xs:restriction>
</xs:simpleType>
</xs:attribute>
<xs:attribute name="DistType" use="required">
    <xs:simpleType>
        <xs:restriction base="xs:int">
            <xs:minInclusive value="0"/>
            <xs:maxInclusive value="18"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>
<xs:attribute name="Subject" type="xs:string" use="optional" default="XmlDocument"/>
<xs:attribute name="DocSize" type="xs:long" use="required"/>
<xs:attribute name="PrintLanguage" use="required">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:enumeration value="D"/>
            <xs:enumeration value="F"/>
            <xs:enumeration value="I"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>
<xs:attribute name="TrustCenterEAN" use="optional">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:pattern value="\d{13}"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>
<xs:attribute name="IsPaperInvoice" use="optional">
    <xs:simpleType>
        <xs:restriction base="xs:boolean">
            <xs:pattern value="true|false"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>
<xs:attribute name="SchemaID" type="xs:string" use="optional"/>
</xs:complexType>
<xs:element name="DocumentsToSend">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Document" type="DocumentType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:schema>

```

3.2 Receiving data

3.2.1 General

- Files on MediPort are retrieved and saved in an input directory.
- Files are given a unique technical file name.
- The system checks that the MediData EAN 7601001304307 is set as intermediary. If not, the intermediary is changed appropriately in the xml and a WARN-log-entry is added.
- Files received are recorded in a log file.
- Files received can be transformed automatically with configurable xslt files.

The receive process can be triggered as follows:

- Communicator runs as an application (all operating systems), service (Windows) or Deamon (Unix/Linux) and the built-in scheduler triggers the receive process automatically. The scheduler can be configured as a basic interval scheduler or as a cron scheduler.
- Communicator runs as an RMI server (all operating systems) and the receive process is triggered externally.

Peculiarities of XML4.3 and eKARUS documents:

- Upon receipt, the document is saved with exactly one transport / via element containing the MediPort EAN 7601001304307. If necessary, the received document is modified accordingly. For generalContainer_430.xsd documents, this applies to the invoice and the MCD.

3.2.2 Multi User

Multi-user functionality is available in MPC V1.5.0 and above and is not just restricted to receiving. Further information is found in chapter 3.8.

3.2.3 XSLT Transformation

Per client and type (xsd schema) of the received file an xslt file can be configured, with which a received file is transformed. The transformed file gets the name of the received file which is not stored. The assignment of a schema and xslt file to be used for that schema is to be done by using an accompanying integer-number m , so for each client and document type 2 properties as follows are to be configured (for the rest, the configuration uses the same principles as for the other parameters per client, as in chapter 5.2.6):

mpcommunicator.config (optional)		
Key	Example Value	Default
CLIENT. n .XSLT. m .DOCTYPE	generalInvoiceResponse_430.xsd	
CLIENT. n .XSLT. m .XSLT	data/xslt/generalInvoiceResponse_430.xslt	

Notes:

- Configured schemas are stored with small letters, so the use of capital or small letters in the schemaname in a received document is irrelevant
- The `CLIENT.n.XSLT.m.DOCTYPE` property shall only contain the filename of the schema (as in the example, without schemaLocation `http://` etc.), the `CLIENT.n.XSLT.m.XSLT` property shall also contain the relative or absolute path including the xslt filename
- If need be, the automatic deletion of received files can be disabled with `mpcommunicator.receive.xslt.deleteorig=false`, then the received document and the transformed document have the same filenames except for the last numbers before the suffix (timestamp).

3.2.4 eKARUS to XML 4.3 conversion

The use of this function must be authorised by MediData.

The following conversions are possible upon receipt of a document:

- `ekarus_x230.xsd` → `generalInvoiceRequest_430.xsd`
- `ekarus_x240.xsd` → `hospitalMCDRequest_430.xsd`
- `generalContainer_430.xsd` with `ekarus_x230.xsd` and/or `ekarus_x240.xsd` → `generalContainer_430.xsd` with `generalInvoiceRequest_430.xsd` and/or `hospitalMCDRequest_430.xsd`

The following parameters must be configured. As for the receive directories per client (chapter 5.2.6), extra directories per client must be specified for the original eKARUS documents, and these directories must be different from the other directories, which contain the converted documents. In addition, this function must be enabled:

mpcommunicator.config (optional)		
Key	Definition	Default
<code>CLIENT.n.RECEIVEORIG_DIR</code>	Receive directory original productive documents	
<code>CLIENT.n.RECEIVEORIGTEST_DIR</code>	Receive directory original test documents	
<code>mpcommunicator.receive.ekarusconversion.enable</code>	eKARUS conversion: true or false	false

3.3 Address book

3.3.1 General

- Details of communication partners are loaded by MediPort and saved in a text file.
- The contents of this file must be evaluated by the vertical application in order to prevent faulty XML documents (unknown recipients).
- The time interval at which the address book is updated can be set in the configuration file.

Updating of the address book can be triggered as follows:

- Communicator runs as application (all operating systems), service (Windows) or Deamon (Unix/Linux) and the built-in scheduler triggers the receive process automatically. The scheduler can be configured as a basic scheduler or as a cron scheduler.
- Communicator runs as an RMI server (all operating systems) and address book updating is triggered externally.

Note:

From MPC Version 1.9.0 on the partner address book is automatically rewritten correctly when the MediPort server is changed (property mpcommunicator.mediport.ip). (Previously, the file config/db/mpcommunicator.script had to be replaced with the default file in the directory config/default/db/).

3.3.2 Address book set up

The address book is set up as follows:

<Header>

<Entry1>

<Entry1>

...

<Entryn>

Example of an address book

```
EAN;ORGANISATION; DEPARTMENT;  
7601000173003;Concordia; Confidential medical service;  
7601003000078;Suva;Test;
```


3.4 Status tracking

3.4.1 Function description

The MPC scheduler initialises the 'GetDocumentList' task at regular intervals on the basis of the configuration. The MPC notes the update time. The server response is saved in a specific directory in XML file format with a relevant file name. The vertical application (VA) processes the new received files at regular intervals and deletes them once they have been processed. Processing by the VA essentially involves updating the status.

3.4.2 Return file

Storage location and name allocation

The server response is saved in file format in the <INSTALL_DIR>/data/docstatus directory. File names are created according to the following schema:

Name schema:

ML_YYMMDD_yymmddhhmmss_NNN.XML where:

- ML File identification
 - YYMMDD Retrieval start date
 - yymmddhhmmss Retrieval end date (with details of time)
 - NNN Consecutive numbering in the event of document statuses in excess of 1,000
-
- XML File extension

3.4.3 XML schema

Contents are saved according to the following XML schema.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema      xmlns:xs="http://www.w3.org/2001/XMLSchema"      elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <xs:element name="GetDocListResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="Document" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="Document">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="AugmentedDocs" type="AugmentedDocInfo" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="DateReceived" type="xs:string" use="required"/>
      <xs:attribute name="MDID" type="xs:unsignedLong" use="required"/>
      <xs:attribute name="DocStatus" type="xs:byte" use="required"/>
      <xs:attribute name="StatusDate" type="xs:string" use="required"/>
      <xs:attribute name="SenderDocumentID" type="xs:string" use="optional"/>
      <xs:attribute name="Subject" type="xs:string" use="optional"/>
      <xs:attribute name="DocSize" type="xs:unsignedLong" use="optional"/>
      <xs:attribute name="Role" type="xs:string" use="optional"/>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="AugmentedDocInfo">
    <xs:attribute name="Receivertyp" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:NMTOKEN">
          <xs:enumeration value="TrustCenter"/>
          <xs:enumeration value="Patient"/>
          <xs:enumeration value="Partner"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="Receiveraddress" type="xs:string" use="optional"/>
    <xs:attribute name="Status" type="xs:byte" use="required"/>
    <xs:attribute name="MDID" type="xs:unsignedLong" use="optional"/>
  </xs:complexType>
</xs:schema>
```

Document attribute list

Parameter	Definition	Indicator	List of values
DateReceived	Send date	"DD.MM.YYYY – HH:MM"	
MDID	MediPort internal identification	Ten digit number	Ten digit number with 1xxxxxxxxx

DocStatus	Document status		According to status table
StatusDate	Status change time stamp	“DD.MM.YYYY – HH:MM”	
SenderDocumentID	ID of sender (usually invoice number)	Must be unique to each sender	
Subject	Description of sender	Free text	
DocSize	Document size	In bytes	
Role	Test or productive		Test or productive

AugmentedDocInfo attribute list

Parameter	Definition	Indicator	List of values
Receivertyp	Indicates the partner type to which copy is sent	Partner is inactive and for future extensions	Trustcenter / Patient / Partner
Receiveraddress	Recipient's EAN number		EAN of Trustcenter EAN MediPort Trustcenter
Status	Document status		According to status table
MDID	MediPort internal ID for copy	Is not currently sent back	Ten digit number

Status table

Document status codes (Range 1 to 99)		
Code	Description	Description
1	Arrived in MediPort server inbox	Document has been received and MediPort has confirmed receipt
2	Being processed	Document is being processed as standard
3	TG processing complete not collected	Document processing in Tiers Garant system completed and document ready for collection
4	Cancelled	Document cancelled by sender
5	Processing complete and collected	Document has been processed (picked up electronically by recipient or printed)
98	Not deliverable via print output	Document could not be printed. Usually delivery address is missing
99	Not deliverable	Document could not be delivered as requested

3.5 Sending without send control files

3.5.1 Functional overview

From version 1.9.0 on the MPC can send without requiring send control files to be provided (chapter 3.1.2). Instead, the desired parameters can be specified, or the MPC uses default values, then the MPC writes the send control files by itself.

Instead of placing the xmls to be sent in the send directory of a client, these xmls are now to be placed in other configured directories. For each such directory, accompanying parameters can be defined, according to which the MPC then automatically generates the corresponding send control files.

The generation of send control files can also be controlled directly by some generic task, i.e. plugin (chapter 3.7), which thus may create send control files more flexibly than preconfigured, e.g. with additional information about a schemaID available only at runtime.

3.5.2 Configuration of the parameters

In the configuration file `mpcommunicator.config` one can specify any number of parameter-sets per client, for which the MPC then automatically generates the corresponding send control files.

Note:

- The parameters of one parameter-set must have identical values `CLIENT.n.DIR.m` in the key, where *n* and *m* are arbitrary (positive) integer values and *n* identifies the corresponding `CLIENT` (specified in `CLIENT.n.EAN`) and *m* identifies the parameter-set.
- A parameter-set must at least specify the first parameter `CLIENT.n.DIR.m`. This parameter defines the directory where the xmls to be sent are to be placed. This path may be relative (cf. example below) or absolute (e.g. `CLIENT.50.DIR.100=D:/data/mpctests/testclient`). Xmls that are incomplete, because e.g. the filesystem is in the process of writing the file, are ignored. As the send control file only permits xmls with the suffix ".xml", xmls with the suffix ".XML" or ".Xml" are changed to ".xml". Other files are ignored.
- Important: If the `smb://` protocol is already used for the configuration of the send, receive, error and archive directories, then the parameter `CLIENT.n.DIR.m` must also be configured as smb directory (cf. chapter 5.3.3).
- If the facultative parameter `CLIENT.n.DIR.m.PLUGIN` is used, then this parameter-set-configuration is ignored by this MPC function and it is only processed by some generic task of the specified name.
- The other 7 parameters are facultative and specify the attributes of the same name used in the send control files, described in chapter 3.1.2, and their values must also be set accordingly.

- If facultative parameters are not configured, the MPC uses default values for `DOCPRINTED`, `DISTTYPE` und `PRINTLANGUAGE` as shown below, and for the other parameters the MPC does the following:

- `DOCATTR`: For all document types in the table below, where **DocAttr direct** is specified, `DocAttr` is automatically set to `direct`, irrespective of any configuration. For all other document types, if a value is configured, it takes precedence, otherwise the xml is parsed: if the xml contains an element `invoice:tiers_payant` then `DocAttr` is set to `Tiers_Payant`, or if the xml contains an element `invoice:tiers_garant` then `DocAttr` is set to `Tiers_Garant_Manuell`. If `DocAttr` cannot be determined `DocAttr` is set to `direct`.

Correct `DocAttr` values for EBPP and XML4.3 documents:

Document type	Constraint	DocAttr
EbppInvoiceRequest_100.xsd		direct
EbppAdminMessage_100.xsd		direct
generalContainer_430.xsd	contains generalInvoiceRequest with tiers_payant element (with or without MCD)	Tiers_Payant
generalContainer_430.xsd	contains generalInvoiceRequest with tiers_garant element (with or without MCD)	Tiers_Garant_Manuell
generalContainer_430.xsd	contains generalInvoiceRequest without tiers_payant or tiers_garant element (with or without MCD)	ERROR
generalContainer_430.xsd	contains generalInvoiceResponse (with or without MCD)	direct
generalContainer_430.xsd	contains only MCD	direct
generalContainer_430.xsd	empty (neither invoice nor MCD)	ERROR
generalCreditRequest_430.xsd		direct
generalCreditResponse_430.xsd		direct
generalFormRequest_430.xsd		direct
generalFormResponse_430.xsd		direct
generalInvoiceRequest_430.xsd	contains tiers_payant element	Tiers_Payant
generalInvoiceRequest_430.xsd	contains tiers_garant element	Tiers_Garant_Manuell
generalInvoiceRequest_430.xsd	contains neither tiers_payant nor tiers_garant element	ERROR
generalInvoiceResponse_430.xsd		direct
generalNotification_430.xsd		direct
hospitalMCDRequest_430.xsd		direct
hospitalMCDResponse_430.xsd		direct
statusRequest_430.xsd		direct
statusResponse_430.xsd		direct

- `TRUSTCENTEREAN`, `ISPAPERINVOICE`, `SCHEMAID`: As these parameters are facultative in the send control file as well, they are dropped.

mpcommunicator.config (optional)		
Key	Example value	Default
CLIENT.n.DIR.m	data/client1/invoices1	
CLIENT.n.DIR.m.DOCATTR	Tiers_Payant	direct
CLIENT.n.DIR.m.DOCPRINTED	false	true
CLIENT.n.DIR.m.DISTTYPE	1	0
CLIENT.n.DIR.m.PRINTLANGUAGE	I	D
CLIENT.n.DIR.m.TRUSTCENTEREAN	1234567890123	
CLIENT.n.DIR.m.ISPAPERINVOICE	false	
CLIENT.n.DIR.m.SCHEMAID	26020	
CLIENT.n.DIR.m.PLUGIN	HL7	

The remaining parameters required for the send control file are automatically set by the MPC (SenderDocID and Subject are set as FileName without suffix).

3.5.3 Configuration of the task

The task is controlled with the following parameters set in the configuration file `mpcommunicator.config` as follows:

Note:

- `mpcommunicator.writesendcontrol.pollinterval` and `mpcommunicator.writesendcontrol.cron` are to be set in the same way as the corresponding parameters of the standard tasks, and their usage is analogous (i.e. only set one of the two, else `mpcommunicator.writesendcontrol.cron` has priority)

mpcommunicator.config (optional)		
Key	Definition	Default
<code>mpcommunicator.task.writesendcontrol.enable</code>	<code>true</code> enables the task, <code>false</code> disables it.	<code>false</code>
<code>mpcommunicator.writesendcontrol.pollinterval</code>	Execution-interval of this task in minutes.	5
<code>mpcommunicator.writesendcontrol.cron</code>	Cron for this task.	

If the task is enabled, the MPC reads all directories specified with `CLIENT.n.DIR.m` regularly, writes the send control files in the clients' send directories according to the corresponding parameter sets and also moves the xmls to be sent to these send directories. There the files are processed by the send task as usual.

If RMI operation is enabled (chapter 5.2.4), the task can be executed manually by using the command

MPC_RMI_Write_Send_Control_Start.exe

3.6 Archiving

From version 1.9.0 on the MPC can archive sent xmls along with their send control files.

For each client an archive directory can be specified in the configuration file. There, for every processed send control file a new directory *yyyy-MM-dd_HH-mm-ss_S* is created, which shows the current processing time in ms-precision S. In each such directory a send control file with all its successfully sent xmls is stored.

A separate task deletes archives that are older than a given age.

mpcommunicator.config (optional)		
Key	Bedeutung	Default
CLIENT. <i>n</i> .SENDARCHIVE_DIR	Archive directory	CLIENT. <i>n</i> .SEND_DIR /archive
mpcommunicator.task.archive.enable	true enables the task, false disables it	false
mpcommunicator.archivedelete.pollinterval	Execution-interval of the task that deletes archived xmls, in hours	48
mpcommunicator.archivedelete.cron	Cron for the deletion of archived xmls	
mpcommunicator.archivedelete.maxtime	Number of days before archived xmls are deleted, if the task is enabled	365

Notes:

- Archiving and deletion of the archives is enabled and disabled simultaneously with the same property. The exception is the manual deletion of archives per RMI with the command `MPC_RMI_Delete_Old_Archives_Start.exe` which is always operational when RMI is enabled.
- If a send control file lists xmls of which some were sent successfully and others produced errors, the send control file is put both in the corresponding archive directory (along with the successfully-sent xmls) and the error directory (along with the xmls not sent successfully).
- If the MPC is stopped while a send control file has only been processed partially and the MPC is restarted again later, a new archive directory is created after the restart for the part of the send control file that had not been processed yet, and the send control file is put in the first and the new archive directory.
- If an archive directory does not contain any successfully sent xmls, it is deleted again automatically.
- As the send directories also the archive directories can be configured with the `smb://` protocol (as they are subdirectories of the former by default) (chapter 5.3.3). Important: If the `smb://` protocol is to be used, then all

directories which are configurable for use with the `smb://` protocol must also be configured to do so, not only some of them.

3.7 Generic Tasks

3.7.1 Functional overview

From MPC version 1.9.0 on any number of arbitrary user-written tasks can be executed by the MPC. This "plugin" functionality can be used e.g. to create standard xml invoices from proprietary data, to easily create send control files, to further process received responses etc.

Generic tasks can be used just like the standard tasks with the interval scheduler, with the cron scheduler or when running the MPC as a windows service, however, a manual operation via RMI is currently not supported. Furthermore, generic tasks cannot be configured per client (if required, the user can easily implement such dependencies himself).

3.7.2 Implementing generic tasks

The java class that is in control must extend `SkipableJob` (contained in `MPCCommunicator/lib/MPCCommunicator.jar`) and implement the method `execute_internal()`, which is then executed as configured by the task scheduler in the MPC:

```
public class MyPlugin extends
com.medidata.mediport.communicator.scheduler.SkipableJob
public void execute_internal() { ... }
```

Note:

The java source level of the added classes must be compatible with the JVM version used. By default, for most operating systems a JVM version 1.6 is installed in the directory `MPCCommunicator/jre/`. Of course, it can be replaced with another JVM, or paths may be adjusted appropriately.

3.7.3 Configuration of generic tasks

In the configuration file `MPCCommunicator/config/mpcommunicator.config` properties must be defined per generic task.

Note:

- *n* is a placeholder for the number of the task. This numbering must begin with 1 and when using several generic tasks they shall be numbered consecutively (1,2,3,...).
- `extension.n.pollinterval` and `extension.n.cron` are to be set in the same way as the corresponding parameters of the standard tasks, and their usage is analogous (i.e. only set one of the two, else `extension.n.cron` has priority).
- Configured generic tasks are automatically enabled (i.e. to disable the task its `extension.n.job` parameter can be commented out).

mpcommunicator.config (optional)			
Key	Description	Example	Default
extension.n.job	Name of the java class which gets control periodically.	ch.md.mp.mpc.ext.mycompany.MyPlugin	
extension.n.pollinterval	Execution-interval of this task in minutes.	5	10
extension.n.cron	Cron for this task.		

Furthermore, all paths (java classpaths) of all start files used must be augmented with the jars/classes of the newly-added tasks:

- Linux: edit `MPCCommunicator/MPC_Scheduler_Start.lax` (augment `lax.class.path`)
- Windows: edit `MPCCommunicator/MPC_Scheduler_Start.lax` (augment `lax.class.path`), for console operation edit `MPCCommunicator/bin/StartMPCCommunicator.bat`, for windows service edit `MPCCommunicator/config/wrapper.conf` (add `wrapper.java.classpath.n`)

3.8 Multi-user functionality

3.8.1 Description of function

Multi-user support is available in MPC version 1.5.0 and above. All standard functions (sending, receiving, docstatus, address book) can be configured individually for all users. From version 1.9.0 on any number of users can be specified using arbitrary (positive) integer numbers (instead of previously writing `CLIENT.1.EAN=...` one can now specify e.g. `CLIENT.1000000.EAN=...`, then `CLIENT.2000000.EAN=...` etc. - all configuration parameters using the same number *n* in `CLIENT.n.XYZ` are considered as belonging to the same `CLIENT` - the old Parameter `CLIENTS.MAXNUMBER` is not used anymore). Users are processed consecutively per task.

Note:

- MediData must have activated the various EAN numbers of the individual clients for the SSI certificate (key store) used by MPC.
- From MPC version 1.9.0 on, clients may share the same send-directories.
- Billing is per client (per EAN number respectively).

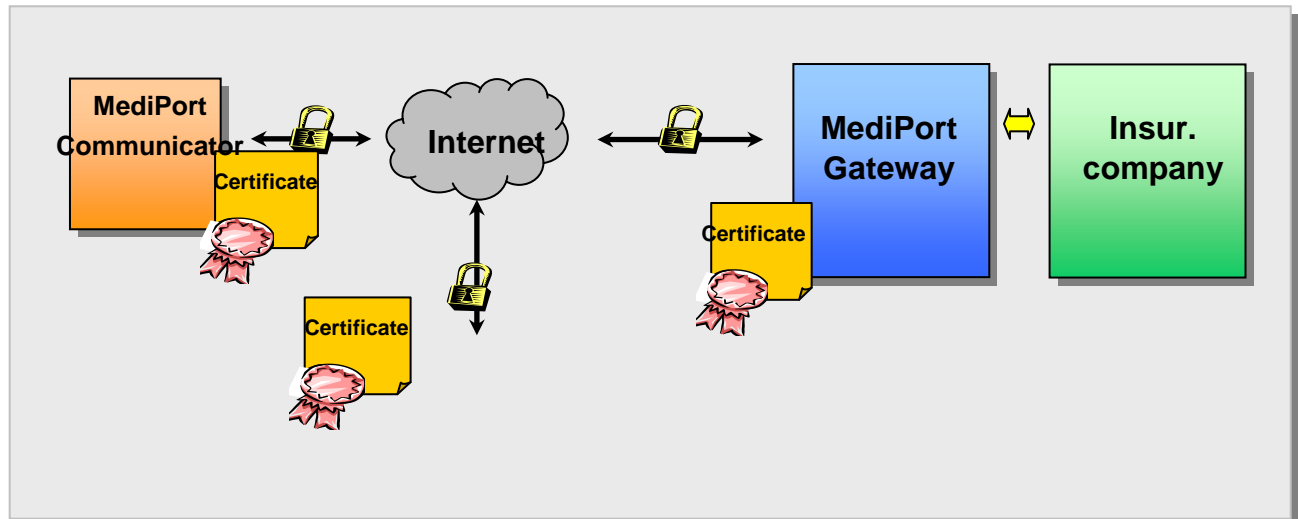
3.9 Security functions / certificates

- A certificate store in the form of an encrypted file is used
- The certificate store contains certificates for the current year and the passwords for reading the certificates
- The certificate store must change at the beginning of each year

Security functions include all functions in connection with the **secure** transfer of documents from and to MediPort. These include authentication, encryption and certificate handling.

3.9.1 Authentication and encryption

Connections with the MediPort server are always made using the SSL process with authentication of **client and server** and the encryption of files during transfer. This technology ensures that the communication partners are authorised and that parties not involved in the process cannot view the data.



3.9.2 Certificate handling

MediData AG supplies a CD with a key store for communication with MediPort that contains valid certificates for the respective year.

3.10 Process control

3.10.1 Timer / Task Manager

- A timer controls the activation of three different tasks:
 - Sending data
 - Receiving data
 - Updating the address book
- Time intervals are defined in the configuration file
- Tasks are not executed if a similar task is still being performed
- In the event of communication problems with the MediPort server tasks will be interrupted and continued at the next specified time.

3.10.2 External trigger

The MPC can also be activated via external triggers instead of via the timer/task control function.

3.11 Logging

3.11.1 Log message layout

- A log file is the interface to the user or user programme
- The log level can be set from INFO to FATAL ERROR
- A log file entry is structured as follows:
 - Time stamp (23 characters)
 - Space (1 character)
 - NDC (nested diagnostic context) per task (S=Send / R= Receive / U=PartnerUpdate / M=Main task / F=FileStop / C=ConsoleStop) (1 character)
 - Space (1 character)
 - Log level (5 characters, filled out with spaces)
 - Space (1 character)
 - (1 character)
 - Space (1 character)
 - Unique identifiable server message (see Appendix B)

Example of a log file entry

```
2003-03-12 09:49:26,906 P INFO - I2902: ADRBOOK UNCHANGED
```

A more detailed example of an MP Communicator log file is given in the appendix.

The configuration is described in the chapter entitled 'Configuration'.

4 Installation

4.1 General

4.1.1 Windows 7 / Windows Vista requirements

For a windows installation, the fact that since Windows Vista program and user files may generally not share the same path anymore has to be considered in particular. The application itself is installed in a folder <INSTALLDIR>, typically C:\Program Files\MPCCommunicator\, however this folder usually remains write-protected for users without administrator rights. In order for non-administrators to be able to operate the MPCCommunicator, a separate configuration and data folder <DATADIR> such as D:\mpcdata\ is used, where all user-configurable files and other files written during operation such as sent/received xmls etc. are put. As the MPCCommunicator may be configured and run by various user accounts, for the sake of simplicity a single user-independent <DATADIR> is used. If a folder for which the current user did not have sufficient rights were used as <DATADIR>, Windows would automatically re-route all write-accesses to another so-called "VirtualStore" folder, which could lead to much confusion. For this reason, during installation a folder without special protection such as D:\mpcdata\ shall be specified as <DATADIR> during installation, for which all potential user-accounts have write access. The installation itself must typically be performed as Administrator (right mouse button-"Run as Administrator").

4.1.2 Initial installation

- Launch the install.bin / install.exe installer direct from installation CD in the 'installer' directory. If you prefer to work with the console version instead of the GUI installer you can start it using the "-i console" option. A program installation folder <INSTALLDIR> and a data installation folder <DATADIR> must be chosen. Except for newer Windows versions (see above), these two folders may also be identical.
- Important: <DATADIR> should absolutely be chosen on a local drive, otherwise neither the log nor some system files can be written during a network interruption and a problem may not be identified!
- Make a new key store file (on the certificate CD, supplied separately) available:
 - Copy the key store file to
<DATADIR>/config/mpcommunicator.keystore
- Configure MPC according to the instructions given in the chapter entitled 'Configuration'.

4.1.3 Directory structure after installation

<INSTALLDIR> e.g.MPCCommunicator/

- bin: Contains files for starting/stopping/testing the application and installation/uninstallation as service (Windows only)
- config: Contains fix and default configuration files
- Documentation: Contains product documentation
- jre: Contains Java run-time environment
- lib: Contains all required run-time libraries
- test: Contains data required for testing the installation
- xsd: schema files
- Wrapper.exe (Windows only): Windows service wrapper
- MPC_xy Start and MPC_xy Stop programs

<DATADIR> e.g. /opt/mpcdata/ or D:\mpcdata\

- config: contains user-configurable files and certificate key store.
- data: This directory contains all files processed by MP Communicator by default (Send/Receive/Address Book) and the log files created
 - send: Contains data to be sent
 - archive: Contains archived sent xmls
 - receive: Contains productive data received
 - test: Contains test data received
 - error: Contains data with a send error
 - log: Contains log file and backups
 - partner: Contains the address book
 - docstatus: Contains document status lists
- test: Contains log file written when testing the installation

4.2 Windows specific

Please refer to the general section on installation.

4.2.1 Installation as Windows service (Optional)

- Execute <INSTALLDIR>/bin/InstallMPCCommunicatorServiceWrapper-Win.bat
Important: Newer Windows versions such as Windows 7 typically only permit services to be installed by the Administrator. For this, execute the file with the right mouse button "Run as Administrator".

Note:

- The service will initialise next time Windows is started. Alternatively the service can be started manually via the Windows services menu. The service is called 'MediData MediPort Communicator'.
- The user used for the service must already have the configured paths (e.g. drive mappings).

4.3 Linux / Unix specific

Please refer to the general section on installation.

4.3.1 Linux / Unix Demon start script mpc.sh

This script supports the following options:

- start : Starts MPC as Demon
- stop : Stops MPC Demon
- console : Starts MPC in current shell
- restart : Restarts MPC Demon
- status : Restores MPC status

4.3.2 Installation as Linux / Unix Demon (Optional)

Creation of a symbolic link in /etc/init.d directory (as root and with entire path):

```
ln -s <INSTALLDIR>/mpc.sh /etc/init.d/mpcommunicator
```

Creation of a symbolic link in the individual run levels:

E update-rc.d mpcommunicator start 20 2 3 4 5 . Stop 20 0 10 6 . or manually:

```
ln -s /etc/init.d/mpcommunicator /etc/rc0.d/K20mpcommunicator
```

```
ln -s /etc/init.d/mpcommunicator /etc/rc1.d/K20mpcommunicator
```

```
ln -s /etc/init.d/mpcommunicator /etc/rc6.d/K20mpcommunicator
```

```
ln -s /etc/init.d/mpcommunicator /etc/rc2.d/S20mpcommunicator
```

```
ln -s /etc/init.d/mpcommunicator /etc/rc3.d/S20mpcommunicator
```

```
ln -s /etc/init.d/mpcommunicator /etc/rc4.d/S20mpcommunicator
```

```
ln -s /etc/init.d/mpcommunicator /etc/rc5.d/S20mpcommunicator
```

5 Configuration

5.1 Minimal configuration

When installing the MediPort Communicator the following adjustments of the default settings must be made in the
<DATADIR>/config/mpcommunicator.config configuration file:

mpcommunicator.config (minimal)		
Key	Definition	Description
mpcommunicator.sender.ean CLIENT.1.EAN	MediPort Communicator Sender EAN	The data required here is on the installation CD in the certificate folder in the xyz.re file
mediport.dn	MediPort Communicator Sender DN	The data required here is on the installation CD in the certificate folder in the xyz.re file
mpcommunicator.mediport.ip	MediPort host name or host address	The IP address of the MediPort host must be changed depending on test/productive operation Productive server : 212.243.92.201 Test server : 212.243.92.199

Example:

```
#=====
# MINIMAL CONFIGURATION
#=====

# MediPort Communicator Sender EAN
mpcommunicator.sender.ean=2099988870017

# MediPort Communicator Sender DN
mediport.dn=uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=insurance company,o=medidata.ch

# MediPort Host Name or Host Address
# Produktiver Server: 212.243.92.201
# Test Server : 212.243.92.199
mpcommunicator.mediport.ip=212.243.92.201

CLIENT.1.EAN=2099988870017
```

All send and receive directories will be correctly set and will not need to be adjusted further.

5.2 Optional configuration

The following configuration settings can be set as an option in the <DATADIR>/config/ mpcommunicator.config configuration file:

5.2.1 Interval scheduler

Minimum values for the intervals are:

- PartnerUpdate : 60 [Minutes]
- Send : 5 [Seconds]
- Receive : 10 [Minutes]
- getdocstate : 10 [Hours]

Note:

- If these minimum values are not adhered to in the configuration, the minimum values shown will be applied automatically.
- The interval scheduler will only be active if 'rmi.enable=false' is set in the configuration file (default setting).

Interval scheduler mpcommunicator.config (optional)		
Key	Definition	Default
mpcommunicator.partner.updateinterval	MPC Address Book update interval in minutes.	240
mpcommunicator.send.pollinterval	MPC Send poll interval in seconds.	10
mpcommunicator.receive.pollinterval	MPC Receive poll interval in minutes.	60
mpcommunicator.getdocstate.pollinterval	MPC Getdocstate poll interval in hours.	10

Example:

```
#####
# TASKS (SIMPLE)
#####
# REMARK: Minimal values for the intervals are:
#   - PartnerUpdate : 60 [minutes]
#   - Send          : 5 [seconds]
#   - Receive       : 10 [minutes]
#   - getdocstate   : 10 [hours]
#####

# MediPort Communicator Partnerverzeichnis update interval[m]
mpcommunicator.partner.updateinterval=240

# MediPort Communicator Send poll interval[s]
mpcommunicator.send.pollinterval=10

# MediPort Communicator receive poll interval[m]
mpcommunicator.receive.pollinterval=60

# MediPort Communicator getdocstate poll interval[h]
mpcommunicator.getdocstate.pollinterval=10
```

5.2.2 Checks before sending

These properties should not be needed or adjusted during normal operation, therefore they have not been included in the mpcommunicator.config file by default. If they are not explicitly set, the default values will be used automatically.

send checks mpcommunicator.config (optional)		
Key	Definition	Default
schemadir	folder with schema files	<INSTALLDIR> /xsd/
maxvalidationlogentries	max. number of validation errors in the log per xml	5
mpcommunicator.send. checkunique.enable	Check multiple sending of the same file, enable/disable (true/false)	true
mpcommunicator.send. checkunique.interval	time span during which hashes of sent xmls are stored for uniqueness check, in minutes	60
uniquedocs	max. number of most recently sent xmls for which the uniqueness check is performed; if within the last hour less xmls were sent, only these are checked	5000
maxhashsize	max. size of an xml in KB for which the complete xml is compared to already-sent xmls; for larger xmls, only the SenderDoc ID is compared	200
deletespercheckpoint	number of xmls after which the internal database is "tidied up"	5000
mpcommunicator.send. checkintermediate.enable	check intermediary of xml, enable/disable (true/false)	true

5.2.3 Send control

The prefix (names of control files used must begin with this character string) of the send control file is determined here. The send control file contains information regarding files sent. The send control file must be located in the corresponding send directory.

Note:

- Depending on the operating system used the name is case sensitive (a difference is made between upper and lower case).
- The send control file actually created may contain any character string after the prefix, but must end in '.xml' (suffix).

Send control mpcommunicator.config (optional)		
Key	Definition	Default
mpcommunicator.sender.controlfile	Sender control file (Case sensitive). Names of the control files used must begin with this character string.	SendControl

Example:

```
#####
# SENDCONTROL
#####

# MediPort Communicator Sender control file (SendControl010_xyz.xml)
# REMARK: !!! Case sensitive!!! SendControl010_ != sendcontrol010_
mpcommunicator.sender.controlfile=SendControl
```

5.2.4 Trigger process control (RMI)

This configuration is required in order to control the MPC using external triggers and not via the built-in scheduler.

Note:

- The RMI port specified is used in each case in V1.4.0 and above in order to prevent the multiple launch of the same MPC instance.

Trigger process control (RMI) mpcommunicator.config (optional)		
Key	Definition	Default
rmi.serverip	IP of the PC on which MPC is running. 'Localhost' can be entered here in virtually every case.	localhost
rmi.serverport	This PC port is used by MPC on start-up.	4445
rmi.servicename	RMI ServiceName. Should not be changed.	MPCCommunicator-Server
rmi.enable	Enables trigger process control and consequently automatic deactivation of scheduler control.	false

Example:

```
#####
# RMI Server
#####
# REMARK: If rmi.enable is set to true then all the
#   tasks are set to false and you can only execute
#   a task by calling its rmi interface.
#####
rmi.serverip=localhost
rmi.serverport=4445
rmi.servicename=MPCCommunicator-Server
rmi.enable=false
```


5.2.5 Proxy configuration

If you do not have direct Internet access, but connection to the Internet only via an (http) proxy server you can configure this proxy server here.

Proxy configuration mpcommunicator.config (optional)		
Key	Definition	Default
proxy.use	Use proxy server (true/false).	false
proxy.ip	Proxy server host name or host address	-
proxy.port	Proxy server port	-
proxy.auth.use	Use proxy server user/password authentication (true/false). Uses proxy server authentication with the user names and passwords shown below.	false
proxy.auth.username	Proxy server authentication user name	-
proxy.auth.password	Proxy server authentication password	-

Example:

```
#####
# Proxy Server
#####
# REMARK: Only "Basic" Proxy Authentication is supported
#####

# Use Proxy Server (true/false)
proxy.use=false

# Proxy Server Host Name or Host Address (192.168.1.122/192.168.3.51)
proxy.ip=1.1.1.1

# Proxy Server Port(8443/8100)
proxy.port=8443

# Use Proxy Server User/Password authentication (true/false)
proxy.auth.use=true

# Proxy Server authentication username
proxy.auth.username=name

# Proxy Server authentication password
proxy.auth.password=password
```

5.2.6 Path configuration

The path configuration has changed in MPC V1.5.0.

Note:

- The corresponding directories must be installed or the MPC must be authorised to create new directories. Note: drive mappings not made cannot be created through MPC.
- Relative paths (default) are interpreted as child folders of <DATADIR> (chapter 4). Absolute paths must start with / (unix systems) resp. *drive_letter*:\ or smb:// (windows). When using "\" in the path, this character must always be used twice. For example, if <DATADIR> was chosen as "D:\mpcdata", then the following 3 configurations are identical:
 - CLIENT.1.SEND_DIR=data/send
 - CLIENT.1.SEND_DIR=D:/mpcdata/data/send
 - CLIENT.1.SEND_DIR=D:\\mpcdata\\data\\send
- x can be any (positive) integer number. All parameters with the same x belong to the same client.

Obsolete:

- The MPC configuration prior to V1.5.0 still continues to function, but may no longer be supported in a later version.

Path configuration mpcommunicator.config (optional)		
Key	Definition	Default
CLIENT.x.EAN	EAN of respective client. MediData must have activated this EAN in respect of the SSL certificate (key store) used.	
CLIENT.x.SEND_DIR	Send directory of respective client.	
CLIENT.x.RECEIVE_DIR	Receive directory for respective client's productive documents.	
CLIENT.x.RECEIVETEST_DIR	Receive directory for respective client's test documents.	
CLIENT.x.ERROR_DIR	Error registry of respective client.	
CLIENT.x.DOCSTAT_DIR	Document status directory of respective client.	
CLIENT.x.PARTNER_FILE	Address book file of respective client	

Example (Standard scenario 1 client):

```
#####
# PATH CONFIGURATION
#####

### NEW MULTI CLIENT (SENDER) FROM V01.05.00
# CLIENT 1 (of 100)
CLIENT.1.EAN=2099988870017
CLIENT.1.SEND_DIR=data/send
```

```
CLIENT.1.RECEIVE_DIR=data/receive  
CLIENT.1.RECEIVETEST_DIR=data/receive/test  
CLIENT.1.ERROR_DIR=data/error  
CLIENT.1.DOCSTAT_DIR=data/docstatus  
CLIENT.1.PARTNER_FILE=data/partner/partnerinfo.txt
```

Example (Special scenario 2 clients)

```
#####  
# PATH CONFIGURATION  
#####  
  
### NEW MULTI CLIENT (SENDER) FROM V01.07.00  
  
# CLIENT 1  
CLIENT.1.EAN=2099988870017  
CLIENT.1.SEND_DIR=data/send  
CLIENT.1.RECEIVE_DIR=data/receive  
CLIENT.1.RECEIVETEST_DIR=data/receive/test  
CLIENT.1.ERROR_DIR=data/error  
CLIENT.1.DOCSTAT_DIR=data/docstatus  
CLIENT.1.PARTNER_FILE=data/partner/partnerinfo.txt  
  
# CLIENT 2  
CLIENT.2.EAN=2099988870017  
CLIENT.2.SEND_DIR=data2/send  
CLIENT.2.RECEIVE_DIR=data2/receive  
CLIENT.2.RECEIVETEST_DIR=data2/receive/test  
CLIENT.2.ERROR_DIR=data2/error  
CLIENT.2.DOCSTAT_DIR=data2/docstatus  
CLIENT.2.PARTNER_FILE=data2/partner/partnerinfo.txt
```

5.2.7 Document status

The status of sent documents can be polled using this function. In order to deactivate this function the getdocstate task must be disabled.

(mpcommunicator.task.getdocstate.enable=false)

Note:

- Task is deactivated by default.
- The time of the last status poll is saved temporarily in a file (DOCS_STATE_OUTPUTDIR/getdocstate.properties).

Document status mpcommunicator.config (optional)		
Key	Definition	Default
DOCS_STATE_INITIAL_DAYS	When executing the getdocstate task all documents are logged that have changed their status over the last DOCS_STATE_INITIAL_DAYS days or since they were last called up	100

Example:

```
#####
# DOCS_STATE
#####

# get all the docs which are done since today - DOCS_STATE_INITIAL_DAYS or
# since the last time the process was started
DOCS_STATE_INITIAL_DAYS=100
```

5.3 Advanced configuration

5.3.1 Cron scheduler

The cron scheduler can also be used instead of the standard interval scheduler. The time at which the respective task should be performed can be stipulated precisely with the help of this scheduler. This trigger handles the full Unix cron syntax (see Appendix C). As soon as the corresponding CronTrigger is defined the normal trigger will be deactivated. CronTriggers can be defined for the following tasks in the <DATADIR>/config/mpcommunicator.config configuration files:

Cron scheduler mpcommunicator.config (advanced)		
Key	Definition	Default
mpcommunicator.partner.update.cron	MPC Address Book update cron	-
mpcommunicator.send.cron	MPC Send cron	-
mpcommunicator.receive.cron	MPC Receive cron	-
mpcommunicator.getdocstate.cron	MPC Getdocstate cron	

Example:

```
# MediPort Communicator Address Book update cron
mpcommunicator.partner.update.cron=0 0/1 10-13 ? * WED,FRI

# MediPort Communicator Send cron
mpcommunicator.send.cron=0 0/1 10-13 ? * WED,FRI

# MediPort Communicator receive cron
mpcommunicator.receive.cron=0 0/1 10-13 ? * WED,FRI

# MediPort Communicator getdocstate cron
mpcommunicator.getdocstate.cron=0 0/1 10-13 ? * WED,FRI
```

5.3.2 Multiple MPC services

In order to allow multiple installations of MPC as a service in Windows, the following procedure must be followed:

The wrapper.conf file in the <INSTALLDIR>/config/ directory must be customised as follows (4 changes):

```
# Port which the native wrapper code will attempt to connect to
wrapper.port=1778
# Name of the service
wrapper.ntservice.name=MediData Second MPC
# Display name of the service
wrapper.ntservice.displayname=MediData Second MPC
# Description of the service
wrapper.ntservice.description=MediData Second MPC
```

5.3.3 SMB File Server Access

From MPC V1.8.0 on the send, receive and error directories, and from MPC V1.9.0 on also the archive directories and the directories for sending without send control files (chapter 3.5) may be on an SMB file server, i.e. they may be remotely accessed Microsoft Windows "share" directories. In the directory path, the smb protocol must be specified, e.g.

```
CLIENT.1.SEND_DIR=smb://dataserver/testuser/data/send
```

Important: If the `smb://` protocol is to be used, then all directories which are configurable for use with the `smb://` protocol must also be configured to do so, not only some of them, i.e. for each client *n* always all 6 properties `CLIENT.n.SEND_DIR`, `CLIENT.n.SENDARCHIVE_DIR`, `CLIENT.n.RECEIVE_DIR`, `CLIENT.n.RECEIVETEST_DIR`, `CLIENT.n.ERROR_DIR` and `CLIENT.n.DIR.m` must be configured with `smb://` (whenever these properties are set). This should not present any problem, because even if some of these directories are located on the local machine, they may nevertheless be configured as Windows "share" directories.

In addition, the appropriate properties, whose names always begin with "jcifs", must be set. In the simplest case, only username, password and perhaps domain need to be set:

SMB file server access mpcommunicator.config (advanced)		
Key	Definition	Default
jcifs.smb.client.username	default username	-
jcifs.smb.client.password	default password	-
jcifs.smb.client.domain	default domain	-
etc.	over 50 properties may be set as needed	-

The list of all possible properties that may be set can be found on
<http://jcifs.samba.org/src/docs/api/>

5.4 Logger configuration

5.4.1 General

Customisation of <DATADIR>/config/mpcommunicator.logger logging configuration file

- Log file (key: log4j.appender.A1.File, default: data/log/mpcommunicator.log)
- Maximum log file size (key: log4j.appender.A1.MaxFileSize, default: 500KB)
- Number of back-up log files until this is overwritten again (key: log4j.appender.A1.MaxBackupIndex, default:3)

5.4.2 Mail (SMTP)

In Version 1.3.x and above issuers have the option to send an e-mail in the event of error. The following procedure must be followed:

The second line of the configuration file must be changed from

```
log4j.rootCategory= INFO, A1, A2
```

to

```
log4j.rootCategory= INFO, A1, A2, SMTP
```

The following must also be set:

- Mail recipient (key: log4j.appender.SMTP.To)
- Mail sender (key: log4j.appender.SMTP.From)
- Mail / Smt server (key: log4j.appender.SMTP.SMTPHost)

Note:

If an e-mail is to be sent to several recipients, these must be separated with a comma (',').

5.4.3 SNMP trap sender

The second line of the configuration file must be changed from

```
log4j.rootCategory=INFO, A1, A2
```

to

```
log4j.rootCategory=INFO, A1, A2, SNMP
```

If errors occur in MPC, the MPC can generate an SNMP message and make this available to the SNMP management server via the network. The following values shown in bold can be configured.

SNMP Trap mpcommunicator.logger		
Key	Definition	Default
log4j.appender.SNMP.ManagementHost	Address of the host to which the SNMP trap should be sent.	-
log4j.appender.SNMP.ManagementHostTrapListenPort	Host port to which the SNMP trap should be sent.	162
log4j.appender.SNMP.EnterpriseOID	EnterpriseOID	-
log4j.appender.SNMP.LocalIPAddress	IP address of the PC on which MPC is installed. This information is then sent in the agent's IP address field to the host.	-
log4j.appender.SNMP.LocalTrapSendPort	Port of the PC from which the SNMP trap should be sent.	163
log4j.appender.SNMP.GenericTrapType	Generic trap type (defined by SNMP Standard). Possible values are: 0 -- cold start 1 -- warm start 2 -- link down 3 -- link up 4 -- authentication failure 5 -- EGP neighbour loss 6 -- enterprise specific	6
log4j.appender.SNMP.SpecificTrapType	Application specific trap type. Value between -128 and 127.	0
log4j.appender.SNMP.ApplicationTrapOID	ApplicationTrapOID	-
log4j.appender.SNMP.CommunityString		public

Example:

```
log4j.appender.A3=org.apache.log4j.ext.SNMPTrapAppender
log4j.appender.A3.ImplementationClassName=org.apache.log4j.ext.JoeSNMPTrapSender
log4j.appender.A3.ManagementHost=192.168.1.99
log4j.appender.A3.ManagementHostTrapListenPort=162
log4j.appender.A3.EnterpriseOID=1.3.6.1.4.1.24.0
log4j.appender.A3.LocalIPAddress=192.168.1.99
log4j.appender.A3.LocalTrapSendPort=163
log4j.appender.A3.GenericTrapType=5
log4j.appender.A3.SpecificTrapType=0
log4j.appender.A3.ApplicationTrapOID=1.3.6.1.4.1.24.2
log4j.appender.A3.CommunityString=public
log4j.appender.A3.SysUpTime=1000
```

5.5 Configuration test

5.5.1 Conditions

- The test will not be performed correctly if MPC has already started. In this case MPC must first be stopped.
- The SenderEAN of the test files in the <INSTALLDIR>/test/testdata/send_receive directory must correspond to the individual EAN. Please replace EAN number 2099988870161 with your EAN number in all .xml files.

5.5.2 Execution

The configuration can be tested by executing '`<INSTALLDIR>/MPC_TestSuite_Start`'. If everything has been configured correctly it should be possible to perform this test without error messages. The logfile of these tests is written to the directory `test/temp/log/`.

6 Update

6.1 Update

As quite a number of files are concerned, it is recommended to uninstall the old version of the MPC (cf. chapter 9) and thereafter to install the new version. For this it may be useful to temporarily save the old configuration file(s) elsewhere, so that these are not overwritten with the default ones during the new installation, so the configuration does not have to be redone from scratch. By simple comparison of the old and new configuration file(s) one can easily see which properties have been added in the new version. Usually only the file `config/mpcommunicator.config` is concerned, and perhaps additionally the file `config/mpcommunicator.logger` if it contains any installation-specific settings.

7 Operation

7.1 General

7.1.1 Start-up

The MediPort Communicator can be started as an RMI server or as an application with a built-in scheduler. Only one of these two functions can run at any given time. Unlike with the scheduler various tasks can be triggered externally with the RMI server.

Scheduler

<INSTALLDIR>/MPC_Scheduler_Start

RMI Server

rmi.enable=true must be set in the configuration file in order to start the RMI server.

<INSTALLDIR>/MPC_RMI_Start

7.1.2 Operation

- Documents collected from the server are stored in the <DATADIR>/data/receive or the <DATADIR>/data/receive/test directory (if test documents are involved)
- You can save XML documents to be sent in the <DATADIR>/data/send directory and then save the associated control file. **Note:** The associated XML schema file must also be saved in this directory and should not be deleted.
- Log files generated are located in the <DATADIR>/data/log directory.
- The address book created is located in the <DATADIR>/data/partner directory.
- The document statuses generated are located in the <DATADIR>/data/docstatus directory.

7.1.3 Stopping

NOTE: stopping the MPCCommunicator can take up to 20 seconds.

Individual tasks will behave as follows in the event of a stop:

- Send: Sending of the current document will be completed and the processing position saved in the send control files to allow continuation from this position on the next start-up.
- Receive: Receiving of the current document will be completed and connection to the server terminated. Documents remaining on the server will be collected from the server on the next start-up.
- PartnerUpdate: Creation of the address book files will be cancelled and a new address book file will be created on the next start-up.

Note: In this case, the address book may not be available briefly (however, it will be copied to PartnerInfo.txt_OLD).

- GetDocstatus: Creation of the document status files will be cancelled and a new document status file will be created on the next start-up.

Scheduler

<INSTALLDIR>/MPC_Scheduler_Stop

RMI Server

<INSTALLDIR>/MPC_RMI_Stop

7.2 Windows specific

7.2.1 Start-up

Console (CMD)

- <INSTALLDIR>/MPCommunicator/bin cd
- StartMPCommunicator.bat

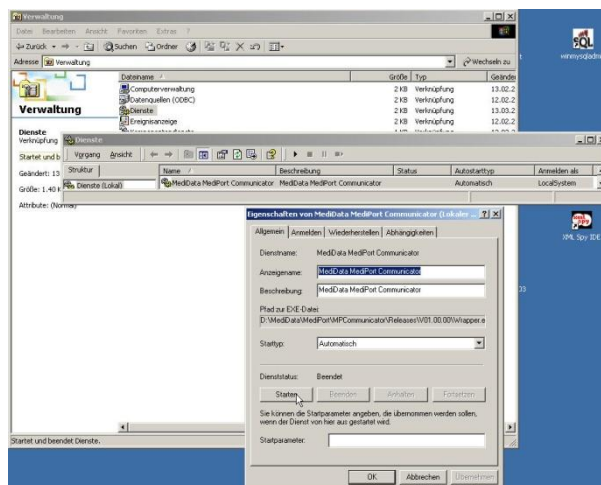
Explorer

Double click <INSTALLDIR>/MPCommunicator/bin/StartMPCommunicator.bat

Windows service

Following installation the 'MediData MediPort Communicator' service will start automatically with the system account each time the computer is started up (even without a user having logged on to the computer).

However, the service can also be started manually via the Windows services programme:



7.2.2 Stopping

Console and Explorer

Two tasks run by default (FileStop in order to be able to stop the application via a stop file and ConsoleStop in order to be able to stop the application via the console) that are responsible for stopping MediPort Communicator:

- By entering "exit" and pressing return in the activated console.
- By creating an "mpcommunicator_stop" file in the <DATADIR>/data directory (File content is irrelevant).

Windows service

Via the Windows services programme on start-up.

8 Maintenance

8.1 Change of certificates

New SSL certificates must be imported once a year.

Copy the keystore from the Certificates CD to the installation directory:

- Make the new keystore file available (supplied separately):
 - Copy the keystore file EANxxxxxxxxxxxxx_mpg.keystore to the directory <DATADIR>/config/ (xxxxxxxxxxxxx is the EAN-number of your certificate)
 - Restart the MPC

Note:

Relevant for the name of the keystore file read by the MPC is the property keystore.name. If the default name as used from version 1.9.0 on

keystore.name=config/EANxxxxxxxxxxxxx_mpg.keystore

is not changed, then from this version on the keystore file need not be renamed to mpcommunicator.keystore anymore. Instead, the MPC looks for a keystore file with the EAN-number as configured in the mediport.dn property. If however one still uses keystore.name=config/mpcommunicator.keystore, then one still has to rename the keystore file correspondingly.

8.2 Application

- Cleaning up log files: Since cyclical log files are saved with a predefined maximum size, it is not necessary to clean up the log files.
- Monitoring/deleting files in the Error directory
- Monitoring/deleting files in the Receive directory
- Monitoring/deleting files in the Docstatus directory

8.3 Database

No special action is required for database maintenance. Since there is no dynamic data in the database apart from the Address Book and the Address Book is updated automatically in the event of a new installation, it is not necessary to make a special back up of the database.

9 Uninstallation

9.1 General

- If MP Communicator has been installed as a Windows service, <INSTALLDIR>/bin/UninstallMPCommunicatorServiceWrapper-Win.bat must be executed.
- Launch the uninstaller in <INSTALLDIR>/UninstallerData/.

Note:

Only files installed by the installer on the hard drive will be removed. Any files added subsequently (e.g. log files, received documents, etc.) will NOT be deleted during uninstallation and need to be removed manually.

10 Troubleshooting

10.1 Problem with Microsoft proxy

Only BASIC authentication is supported (no NTLM authentication).

recently we also came across some problems with authentication towards a Microsoft proxy. The reason was that the proxy was configured to do NTLM authentication and not BASIC authentication.

Unfortunately, NTLM authentication is only supported by Microsoft components such as Internet Explorer (the NTLM protocol is proprietary to Microsoft :-()). The only way to get around this is probably to install the Microsoft WinSock Proxy Client which handles NTLM authentication transparently to any application.

11 Appendix A: File names of received documents

XML file name: **bbbbbbbbbbttttttttttttaiin.XML**

File name encoding:

bbbbbbbbbb Code according to description in table below

ttttttttttt Document first saved on <Time/Date> in
1/1000 seconds since 1.1.1970

a: Document type T = Test; P = Productive

ii: Identification of intermediary sending the document

n: Consecutive number for documents in the same
millisecond

XML File extension for XML documents

Summary of document name codes (Part bbbbbbbbb)

Document	Schema	File name (MPC)
Dentist invoice, Tiers Payant	DentistInvoiceRequest_300.xsd	IRDEP0300
Dentist invoice, Tiers Garant	DentistInvoiceRequest_300.xsd	IRDEM0300
Dentist invoice response	DentistInvoiceResponse_300.xsd	ISDEX0300
Hospital invoice, Tiers Payant	HospitalInvoiceRequest_300.xsd	IRHSP0300
Hospital invoice, Tiers Garant	HospitalInvoiceRequest_300.xsd	IRHSM0300
Hospital invoice response	HospitalInvoiceResponse_300.xsd	ISHSX0300
Invoice reminder	InvoiceReminderRequest_300.xsd	IRREX0300
Invoice reminder response	InvoiceReminderResponse_300.xsd	ISREX0300
Lab invoice, Tiers Payant	LabInvoiceRequest_300.xsd	IRLAP0300
Lab invoice, Tiers Garant	LabInvoiceRequest_300.xsd	IRLAM0300
Lab invoice response	LabInvoiceResponse_300.xsd	ISLAX0300
Doctor invoice, Tiers Payant	MDInvoiceRequest_300.xsd	IRMDP0300
Doctor invoice, Tiers Garant	MDInvoiceRequest_300.xsd	IRMDM0300
Doctor invoice response	MDInvoiceResponse_300.xsd	ISMDX0300
Pharmacy invoice, Tiers Payant	PharmacyInvoiceRequest_300.xsd	IRPAP0300
Pharmacy invoice, Tiers Garant	PharmacyInvoiceRequest_300.xsd	IRPAM0300
Pharmacy invoice response	PharmacyInvoiceResponse_300.xsd	ISPAX0300
Physio invoice, Tiers Payant	PhysioInvoiceRequest_300.xsd	IRPYP0300
Physio invoice, Tiers Garant	PhysioInvoiceRequest_300.xsd	IRPYM0300
Physio invoice response	PhysioInvoiceResponse_300.xsd	ISPYX0300
Shoemaker invoice, Tiers Payant	ShoemakerInvoiceRequest_300.xsd	IRSMP0300
Shoemaker invoice, Tiers Garant	ShoemakerInvoiceRequest_300.xsd	IRSMM0300
Shoemaker invoice response	ShoemakerInvoiceResponse_300.xsd	ISSMX0300
Unspecified invoice, Tiers Payant	UnspecifiedInvoiceRequest_300.xsd	IRUNP0300

Document	Schema	File name (MPC)
Unspecified invoice, Tiers Garant	UnspecifiedInvoiceRequest_300.xsd	IRUNM0300
Unspecified invoice response	UnspecifiedInvoiceResponse_300.xsd	ISUNX0300
"MiGeL" invoice, Tiers Payant	WholesalerInvoiceRequest_300.xsd	IRWSP0300
"MiGeL" invoice, Tiers Garant	WholesalerInvoiceRequest_300.xsd	IRWSM0300
"MiGeL" invoice response	WholesalerInvoiceResponse_300.xsd	ISWSX0300
Unknown document type Tiers Payant	<no description>	UKDTP0000
Unknown document type Tiers Garant	<no description>	UKDTM0000
Dentist invoice 4.0, Tiers Payant	DentistInvoiceRequest_400.xsd	IRDEP0400
Dentist invoice 4.0, Tiers Garant	DentistInvoiceRequest_400.xsd	IRDEM0400
Dentist invoice response 4.0	DentistInvoiceResponse_400.xsd	ISDEX0400
TG Document Request 4.0	DocumentRequest_400.xsd	TGDRX0400
TG Document Response 4.0	DocumentResponse_400.xsd	TGDSX0400
General credit 4.0	generalcreditrequest_400.xsd	CRGEX0400
General credit response 4.0	generalcreditresponse_400.xsd	CSGEX0400
General invoice 4.1, Tiers Payant	generalinvoicerequest_410.xsd	IRGEP0410
General invoice 4.1, Tiers Garant	generalinvoicerequest_410.xsd	IRGEM0410
General invoice response 4.1	generalinvoiceresponse_410.xsd	ISGEX0410
Hospital credit 4.0	hospitalcreditrequest_400.xsd	CRHSX0400
Hospital credit response 4.0	hospitalcreditresponse_400.xsd	CSHSX0400
Hospital invoice 4.0, Tiers Payant	HospitalInvoiceRequest_400.xsd	IRHSP0400
Hospital invoice 4.0, Tiers Garant	HospitalInvoiceRequest_400.xsd	IRHSM0400
Hospital invoice 4.0 response	HospitalInvoiceResponse_400.xsd	ISHSX0400
Invoice reminder 4.0	InvoiceReminderRequest_400.xsd	IRREX0400
Invoice reminder response 4.0	InvoiceReminderResponse_400.xsd	ISREX0400
Lab invoice 4.0, Tiers Payant	LabInvoiceRequest_400.xsd	IRLAP0400
Lab invoice 4.0, Tiers Garant	LabInvoiceRequest_400.xsd	IRLAM0400
Lab invoice response 4.0	LabInvoiceResponse_400.xsd	ISLAX0400
Doctor invoice 4.0, Tiers Payant	MDInvoiceRequest_400.xsd	IRMDP0400
Doctor invoice 4.0, Tiers Garant	MDInvoiceRequest_400.xsd	IRMDM0400
Doctor invoice response 4.0	MDInvoiceResponse_400.xsd	ISMDX0400
Pharmacy invoice 4.0, Tiers Payant	PharmacyInvoiceRequest_400.xsd	IRPAP0400
Pharmacy invoice 4.0, Tiers Garant	PharmacyInvoiceRequest_400.xsd	IRPAM0400
Pharmacy invoice response 4.0	PharmacyInvoiceResponse_400.xsd	ISPAX0400
Physio invoice 4.0, Tiers Payant	PhysioInvoiceRequest_400.xsd	IRPYP0400
Physio invoice 4.0, Tiers Garant	PhysioInvoiceRequest_400.xsd	IRPYM0400
Physio invoice response 4.0	PhysioInvoiceResponse_400.xsd	ISPYX0400
Shoemaker invoice 4.0, Tiers Payant	ShoemakerInvoiceRequest_400.xsd	IRSMP0400
Shoemaker invoice 4.0, Tiers Garant	ShoemakerInvoiceRequest_400.xsd	IRSMM0400
Shoemaker invoice response 4.0	ShoemakerInvoiceResponse_400.xsd	ISSMX0400
Unspecified invoice 4.0, Tiers Payant	UnspecifiedInvoiceRequest_400.xsd	IRUNP0400
Unspecified invoice 4.0, Tiers Garant	UnspecifiedInvoiceRequest_400.xsd	IRUNM0400
Unspecified invoice response 4.0	UnspecifiedInvoiceResponse_400.xsd	ISUNX0400

Document	Schema	File name (MPC)
Wholesaler invoice 4.0, Tiers Payant	WholesalerInvoiceRequest_400.xsd	IRWSP0400
Wholesaler invoice 4.0, Tiers Garant	WholesalerInvoiceRequest_400.xsd	IRWSM0400
Wholesaler invoice response 4.0	WholesalerInvoiceResponse_400.xsd	ISWSX0400
xmit documents 4.0	xmitdocuments_400.xsd	XMGEX0400
xmit documents 4.1	xmitDocuments_410.xsd	XMGEX0410
General container 4.3, Invoice Tiers Payant	generalContainer_430.xsd	GCUKP0430
General container 4.3, Invoice Tiers Garant	generalContainer_430.xsd	GCUKM0430
General container 4.3, only MCD	generalContainer_430.xsd	GCUKX0430
General credit request 4.3	generalCreditRequest_430.xsd	CRGEX0430
General credit response 4.3	generalCreditResponse_430.xsd	CSGEX0430
General form request 4.3	generalFormRequest_430.xsd	FRGEX0430
General form response 4.3	generalFormResponse_430.xsd	FSGEX0430
General invoice 4.3, Tiers Payant	generalInvoiceRequest_430.xsd	IRGEP0430
General invoice 4.3, Tiers Garant	generalInvoiceRequest_430.xsd	IRGEM0430
General invoice response 4.3	generalInvoiceResponse_430.xsd	ISGEX0430
General notification 4.3	generalNotification_430.xsd	NFGEX0430
Hospital invoice MCD 4.3	hospitalMCDRequest_430.xsd	MRHSX0430
Hospital invoice MCD response 4.3	hospitalMCDResponse_430.xsd	MSHSX0430
Status request 4.3	statusRequest_430.xsd	SRUKX0430
Status response 4.3	statusResponse_430.xsd	SSUKX0430
EBPP invoice message	EbppInvoiceRequest_100.xsd	IREBX0100
EBPP administrative message	EbppAdminMessage_100.xsd	AMEBX0100
eKARUS general invoice, Tiers Payant	ekarus_x230.xsd	EKHSP0230
eKARUS general invoice, Tiers Garant	ekarus_x230.xsd	EKHSM0230
eKARUS hospital invoice MCD	ekarus_x240.xsd	EKHSX0240
eKARUS general invoice response	ekarus_x250.xsd	EKHSX0250

Table 1 Creation of file names

See www.forum-datenaustausch.ch for information on current XML schemas.

Summary of file name prefixes

IRDEP0300	1-2: Document type
	IR = InvoiceRequest (invoice)
	IS = InvoiceReSponse (invoice response)
	CR = CreditRequest (credit request)
	CS = CreditReSponse (credit response)
	XM = XmitDocuments (xmit documents)
	MR = McdRequest (minimal clinical dataset)
	MS = McdReSponse (minimal clinical dataset response)
	SR = StatusRequest (status)
	SS = StatusReSponse (status response)

GC = GeneralContainer	(invoice and/or MCD)
FR = medForm Request	(form request)
FS = medForm ReSponse	(form response)
NF = Notification	(notification)
EK = eKARUS	(eKARUS)

IRDEP0300

3-4: Role

DE = D Entist	(Dentist)
HS = H oSpital	(Hospital)
RE = R Eminder	(Reminder)
LA = L Ab	(Lab)
MD = M edical D octor	(Doctor)
PA = P h A rmacy	(Pharmacy)
PY = P h Y sio	(Physiotherapy)
SM = S hoe M aker	(Shoemaker/Orthopaedic technician)
UN = U Nspecified	(Unspecified)
ED = UN/ E DIFACT	(Edifact)
WS = W hole S aler	(MiGeL)
UK = UnKnown	(Unknown type)
GE = G eneral	(general)
EB = E BPP	(EBPP)

IRDEP0300

5: Type of reimbursement

P = Tiers **P**ayant
M = Tiers Garant
X = Irrelevant

IRDEP0300

6-9: Release

0300 = 3.00
0400 = 4.00
0410 = 4.10
0430 = 4.30
0230 = eKARUS InvoiceRequest
0240 = eKARUS MCDRequest
0250 = eKARUS InvoiceResponse
0000 = irrelevant

12 Appendix B: Log messages

12.1 INFO (I0000 – I9999) log level messages

INFO No.	Reason	Display
I0801	A new address book has been created. The old address book has been copied to ..._OLD.	New partnerfile [<filename>] created.
I1001	Processing of a new send control file started.	Start processing new SendControlFile [<sendcontrolfileinfo>]
I2201	Start-up message with details of version	<pre> ***** *** Start MediPort Communicator *** *** *** *** to stop: *** *** exit<ret> or Put mpcommunicator_stop file *** *** into data directory. *** ***** Starting MPCCommunicator V1.0.0 </pre>
I2203	Shutdown message with statistics	RunningSince[<date>] NbrOfJobsExecuted[<nbr>] NbrDocs Sent[<nbr>] NotSent[<nbr>] Received[<nbr>]
I2204	Scheduler prepares tasks	"----- Scheduler schedule jobs -----"
I2205	Info about each task to be performed	grp_mg.<taskname> will run at[<date_of_first_run>] & repeat[<repeat_count>/<repeat_interval_in_ms>] or grp_mg.<taskname> will run at[<date_of_first_run>] with cronExpression[<cron_expression>]
I2206	Scheduler started	"----- Scheduler started -----"
I2207	Scheduler has received stop message and is waiting for all on-going tasks to finish	"----- Shutting Down -----"
I2208	Scheduler stopped. All tasks stopped.	"----- Shutdown Complete -----"
I2401	Document received OK	RECEIVE DOC OK FILENAME[<filename>] SENDERDN[<senderdn>] STATETYPEID[200] MDID[<doc_id>] DOCTYPE[<doc_typ>] JOB[<job_id>]
I2501	Document sent OK	SEND DOC OK FILENAME[<filename>] RECEIVERDN[<receiver_dn>] STATETYPEID[100] MDID[<doc_id>] DOCTYPE[<doc_typ>] JOB[<job_id>]
I2902	PartnerUpdate task has established that the address book on the server has not changed.	ADRBOOK UNCHANGED
I2903	Receive task has established that there are no new documents on the server.	NO NEW DOCUMENTS ON SERVER
I2905	PartnerUpdate task has established that the address book on the <nbr> server contains new/amended entries. However, the new address book on the local hard drive has not yet been updated.	PARTNER changed[<nbr>]

12.2 WARN (W0000 – W9999) log level messages

Warn No.	Display	Reason	Outcome
W0601	Property file not found. Create a new one.	Logging configuration file not found	A new logging configuration file with default values is created.
W0801	< SQLException>	Error when closing Connection/Stmt/ResultSet/File	
W0802	< SQLException>	Error when closing Connection/Stmt/ResultSet/File	
W0803	< SQLException>	Error when closing Connection/Stmt/ResultSet/File	
W0804	SQLWarning: 0 rows INSERTED into table SENDFILEINFO	Error when saving SendControlFile position in DB.	Next time MP Communicator is launched several error messages will appear, as SendControlFile documents that have already been dealt with are no longer there.
W0805	< SQLException>	Error when closing Connection/Stmt/ResultSet/File	
W0806	< SQLException>	Error when closing Connection/Stmt/ResultSet/File	
W0807	SQLWarning: 0 rows Deleted from table SENDFILEINFO	Error when deleting SendControlFile entries in the database.	Warnings on next start-up, since the SendControlFile info in the database are no longer available on the hard drive. If there is already a new SendControlFile on the hard drive the file size and/or the last modified time stamp will no longer be correct and this is why a warning is given.
W0808	< SQLException>	Error when closing Connection/Stmt/ResultSet/File	
W0809	< SQLException>	Error when closing Connection/Stmt/ResultSet/File	
W0810	< SQLException>	Error when closing Connection/Stmt/ResultSet/File	
W0811		SQLException when closing Connection/Stmt/ResultSet	
W0812		SQLException when closing Connection/Stmt/ResultSet	
W0813		SQLException when closing Connection/Stmt/ResultSet	
W0814		SQLException when closing Connection/Stmt/ResultSet	
W0815		SQLException when closing Connection/Stmt/ResultSet	
W0816		SQLException when closing Connection/Stmt/ResultSet	
W0817		New partner has not been entered into the database.	Since it has no FK and no other constraints on the DB, this cannot actually happen.
W0818		SQLException when closing Connection/Stmt/ResultSet	
W0819		SQLException when closing Connection/Stmt/ResultSet	

Warn No.	Display	Reason	Outcome
W0820		Sleep Thread has been interrupted	
W0821		SQLException when closing Connection/Stmt/ResultSet	
W0822		SQLException when closing Connection/Stmt/ResultSet	
W0823		SQLException when closing Connection/Stmt/ResultSet	
W0824		SQLException when closing Connection/Stmt/ResultSet	
W0825		SQLException when closing Connection/Stmt/ResultSet	
W0826		Abbreviation for schema url not found in database. Schema name possibly not entered in DB.	UNKN used as file identifier
W0827		SQLException when closing Connection/Stmt/ResultSet	
W0901		DB connection pool has been increased	
W1010		Desired Input Channel is not a valid input channel	
W1101		Desired Input Channel is not a valid input channel	
W1102		Exception when closing Input / Output stream.	
W1701		Method not impl.	
W2001		Sleep Thread has been interrupted	
W2101		Stop file could not be deleted	If the stop file is still there a relaunch of the application will result in an immediate stop
W2201		Scheduler Exception during statistic output	
W2202		scheduler already started	
W2203		scheduler already stopped	
W2204		Sleep Thread has been interrupted	
W2401		IOException: error when closing in(document) or out(file) streams	
W2501		SQLException: Server has received a stop signal, interrupts the processing of the SendControlFile and tries to write the position in the file into the DB, which is currently not working.	On restart the MiniGateway tries to process the SendControlFile right from the beginning, but cannot find the document already processed and logs several errors. The remaining documents are subsequently processed properly.
W2502		SQLException: Server cannot read the SendControlFile from the DB	See W0601
W2601		One of the jobs has thrown an exception and this is intercepted centrally here.	
W2701		Sleep Thread has been interrupted	
W2901		Unimplemented method HttpClient.isAvailable returns lways false.	
W2902		Exception: closing the JDConnection threw an	

Warn No.	Display	Reason	Outcome
		exception	
W2903		IOException: closing the input stream did not function	
W2904		IOException: closing the input stream did not function	
W2905		IOException: closing the input stream did not function	
W2906		IOException: closing the input stream did not function	
W2907		IOException: closing the input stream did not function	
W2908		IOException: closing the input stream did not function	
W2909		IOException: closing the input stream did not function	
W3601		The CSendFileInfo saved in the DB contains a different file size from the current SendControlFile on the HD.	Processing of the current SendControlFile begins right at the start and does not skip any documents.
W3602		The CSendFileInfo saved in the DB contains a different LastModificationDate from the current SendControlFile on the HD	Processing of the current SendControlFile begins right at the start and does not skip any documents.
W3603		The CSendFileInfo saved in the DB contains a different number of documents from the current SendControlFile on the HD	Processing of the current SendControlFile begins right at the start and does not skip any documents.
W3604		Info about a SendControlFile is saved in the DB, but there is none on the HD at the moment.	No documents are currently being sent.
W4201		laik provider is already registered	
W4202		The certificate on the server is due to be changed.	There can be no communication with the server during this time.
W4203		IOException: output stream could not be closed	
W5801	XML VALIDATION problem[<problem>]	XML validation problem	Documents for sending will be rejected or not depending on the xml.validation.abort.level set.

12.3 ERROR (E0000 – E9999) log level messages

Error Nr	Message	Reason	Outcome	Action
E0801	outputAllPartner failed to move file <temp> to <new>	Renaming of new temp PartnerInfo file to real Partner Info file did not work	Address book is no longer up to date. Next time PartnerUpdate task is launched another attempt will be made to create a new address book.	
E0802	outputAllPartner failed to move file <partnerfile> to <old>	Renaming of old Partnerinfo files to Old PartnerInfoFile did not work	Address book is no longer up to date. Next time PartnerUpdate task is launched another attempt will be made to create a new address book.	

Error Nr	Message	Reason	Outcome	Action
E0803	Close database connection failed.	Closing of database connection failed.	The database connection will eventually break off and the application must be restarted.	
E1001	Can't read SendControlFile[<filename>]	No read rights on SendControlFile	Current processing of SendControlFile is interrupted.	Access rights adjusted
E1002	Error in SendControlFile	NumberFormatException	SendControlFile is not processed, but an attempt is still made to delete it.	
E1003	Error in SendControlFile	XmlParserException	SendControlFile is not processed, but an attempt is still made to delete it.	
E1004	Proper access cannot be gained to the SendControlFile	IOException	SendControlFile is not processed, but an attempt is still made to delete it.	Check access rights.
E1005	File[<filename>] with URL[<url>] not found	The file specified in the SendControlFile does not exist.	This SendControlFile entry is skipped	Manual check to see where this file is located.
E1006	File[<filename>] has not the correct access rights (rw)	File to be processed does not have the necessary access rights.	This SendControlFile entry is skipped.	Manual check to see where this file is located and what access rights it has...
E1007	File[<filename>] has size[<filesize>] but should have size[<filesize>]	File does not have the file size specified in the SendControlFile	This SendControlFile entry is skipped. File is moved to error directory.	Manual check to see why file size is incorrect.
E1008	Building a valid URL failed.	Access to file denied.	This SendControlFile entry is skipped. File is copied to error directory.	
E1009	Reading the input file failed.	Import file failed.	This SendControlFile entry is skipped. File is moved to error directory.	
E1010	Closing of io-streams failed.	Completion of file import failed	This SendControlFile entry is skipped. File is moved to error directory.	
E1101	Document cannot be saved.	Temporary file (TMP extension) in which the received document has been saved cannot be renamed in the final document.	Document is deemed not received and is collected again from the server next time a connection is established. However, the temporary file remains in the input directory.	Manual deletion of temporary file (with TMP extension)
E1102	Target file cannot be generated.	The document received from the server cannot be written to the hard drive.	Document is deemed not received and is collected again from the server next time a connection is established.	
E1103	IO Error during storing information.	The document received from the server cannot be written to the hard drive.	Document is deemed not received and is collected again from the server the next time a connection is established.	
E1201	Document [<docname>] cannot be deleted.	Document to be sent cannot be deleted.	Document is not deleted and remains in the input directory, however this does not necessarily mean that the document has not been sent to the server. In order to confirm this, the preceding messages must be analysed.	Manual deletion of this file.
E1202	move to error channel of document [<docname>] failed.	Transfer of document to error channel failed.	E2503	

Error Nr	Message	Reason	Outcome	Action
E1801	Parsing of XML-content failed.	Error when parsing received document.	Receiving of document being processed interrupted by server and next document collected from the server. The faulty document is collected again from the server next time the receive task is launched.	
E1802	Input source cannot be connected.	Error when receiving document.	Receiving of document being processed interrupted by the server and next document collected from the server. The faulty document is collected again from the server next time the receive task is launched.	
E1803	Stream cannot be closed.	Error when closing received document	Receiving of document being processed interrupted by the server and next document collected from the server. The faulty document is collected again from the server next time the receive task is launched.	
E2301	Try to update partner failed.	Collective error message. Reason see sub-message	PartnerUpdate is interrupted and another attempt is made on next launch of UpdatePartner jobs.	See sub-message
E2401	Using IOInterface failed.	Collective error message. Reason see sub-message	This document could not be read correctly from the server and is consequently skipped. An attempt is then made to read the next document from the server.	See sub-message.
E2402	Receiving document failed.	Collective error message. Reason see sub-message	This document could not be read correctly from the server and is consequently skipped. An attempt is then made to read the next document from the server.	See sub-message
E2403	RECEIVE DOC FAILED <TRANSPORTLOG>	Collective error message. Reason see sub-message	This document could not be read correctly from the server and is consequently skipped. The receipt of documents by the server is interrupted and is only continued the next time the receive task is launched.	
E2404	RECEIVE DOC FAILED	No connection to server	This document could not be read correctly from the server and is consequently skipped. The receipt of documents is only continued the next time the receive task is launched.	
E2501	Failed to delete send control file.	SendControlFile could not be deleted. There is a problem with the access rights.	A new SendControlFile cannot be created.	Check access rights and manually delete SendControlFile
E2502	SEND DOC FAILED <TRANSPORTLOG>	Collective error message. Reason see sub-message.	This document could not be sent to the server and is consequently moved to the error directory.	See sub-message
E2503	Document cannot be sent. Attempt to archive document failed.	Document could not be sent. Attempt to archive document in error directory failed.	Document still located in send directory, but will no longer be sent.	Delete document manually and resend.
E2504	Failed to delete processed document [filename]	Document sent successfully to server. However, subsequent attempt to delete it failed.	Document sent successfully still located in send directory.	Delete document manually.
E2505	Wrong SendControlFile information.	Internal programme error		Inform MediData
E2601	Unimplemented method called	Internal programme error		Inform MediData.
E2901	Try to create partnerInfoFile failed.	Problem with accessing database when creating address	Address book is not recreated. Another attempt is made to create the address book next time the PartnerUpdate	

Error Nr	Message	Reason	Outcome	Action
		book.	task is launched.	
E2902	Try to create partnerInfoFile failed.	Problem accessing the file system when creating address book.	Address book is not recreated. Another attempt is made next time the PartnerUpdate task is launched.	
E2903	Getting document list response failed	Error when collecting new document list from server.	Another attempt to collect new documents from the server is made next time the receive task is launched.	
E2904	Reading document list response failed [Operation timed out: connect]	A network connection to the server could not be established.	During this receive job no more documents are collected from the server. Documents already collected have been processed correctly. A new attempt is made during the next receive job to collect any documents still available from the server.	If this error occurs several times in succession the reason for the network connection problem must be analysed (wrong proxy, server configuration?)
E2905	Parsing document list response failed.	Parsing document list from the server failed.	Next time the receive task is performed another attempt is made to collect new documents from the server.	
E2906	Reading document failed.	Reading document from server failed.	Receiving of this document skipped. Next time the receive task is performed another attempt is made to collect this document from the server.	
E2907	Reading server response stream failed (Operation timed out: connect)	Network connection to the server cannot be established	Processing of documents to be sent is interrupted and the current position saved in the SendControlFile in the DB. Processing will continue from the same place next time the SendControlFile task is launched.	If this error occurs several times in succession the reason for the network connection problem must be analysed (wrong proxy, server configuration?)
E2908	Parsing server response failed.	Error occurred when parsing server response	Depends on the individual server response	
E2909	Building https request failed. Could not parse Sender EAN[<error>] for File[<filename>]	The EAN found in the document to be sent was not a number.	This document could not be sent to the server and is consequently moved to the error directory. An attempt is then made to send the next document from the server.	
E2910	Build https req failed. Could not retrieve data from DB [<error>] for File [<filename>]	Error when accessing the database.	This document could not be sent to the server and is consequently moved to the error directory. An attempt is then made to send the next document from the server.	
E2911	Building https request failed. Could not parse document [<error>] for File[<filename>]	Error when parsing the document to be sent.	This document could not be sent to the server and is consequently moved to the error directory. An attempt is then made to send the next document from the server.	
E2912	Open https-connection failed.			
E2913	Getting address book failed.	Error when parsing the address book from the server	Address book is not recreated. Another attempt is made to create the address book next time the PartnerUpdate	

Error Nr	Message	Reason	Outcome	Action
			task is launched.	
E2914	Getting address book failed [<error>]	Error when parsing the address book from the server.	Address book is not recreated. Another attempt is made to create the address book next time the PartnerUpdate task is launched.	
E2915	Reading address book response failed [<error>]	IOException: Reading address book response failed (Operation timed out: connect)	Address book is not recreated. Another attempt is made to create address book next time the PartnerUpdate task is launched.	
E2916	Parsing address book response failed [<error>]	Error when parsing the address book from the server.	Address book is not recreated. Another attempt is made to create the address book next time the PartnerUpdate task is launched.	
E2918	Posting document failed [<error>]	Document cannot be sent to server	Document remains in input directory.	
E2919	Read address book response failed [<error>]	Reading new address book from server failed	Address book is not recreated. Another attempt is made to create the address book next time the PartnerUpdate task is launched.	
E2920	Parse address book response failed[<error>]	Parsing of new address book from server failed.	Address book is not recreated. Another attempt is made to create the address book next time the PartnerUpdate task is launched.	
E2921	Try to update partner on local database failed[<error>]	Writing of new address book to local database failed.	Address book is not recreated. Another attempt is made to create the address book next time the PartnerUpdate task is launched.	
E2922	Invalid URL[<url>] throws [<error>]	Invalid URL	Document is not received/sent/partner update not performed.	Inform MediData
E2923	Parsing address book response failed[<error>]		Address book is not recreated. Another attempt is made to create the address book next time the PartnerUpdate task is launched.	
E2924	Posting document failed[<error>]	The document cannot be sent to the server.	The document was moved to the error directory.	
E2925	Configured SenderEAN[nnn] doesn't match SenderEAN[nnn] in document[xyz]	Wrong Sender-EAN in the document.	The document was moved to the error directory.	
E2926	Getting getDocumentStates response failed[<error>]	Error during retrieval of document states.		renewed try later on, if one-time error
E2927	Getting getDocumentStates XmlParserException[<error>]	Error during parsing of answer of document states.		renewed try later on, if one-time error
E2928	Reading getDocumentStates response failed[<error>]	Error during retrieval of document states.		renewed try later on, if one-time error
E2929	unknown GetDocListResponse	Error during retrieval of document states.		renewed try later on, if one-time error
E2930	checkServer empty	error during connection setup with server	partner directory is not updated	renewed try later on, if one-time error
E2931	checkServer failed.	error during connection setup with server	partner directory is not updated	renewed try later on, if one-time error

Error Nr	Message	Reason	Outcome	Action
E2932	Reading checkServer response failed[<error>]	error during connection setup with server	partner directory is not updated	renewed try later on, if one-time error
E3501	File[<filename>] has malformedURLException[<error>]	Document URL is invalid.	Document to be sent cannot be read and is moved to error directory.	
E5701	Reading xml document failed[<error>]	Import of XML document failed.	See generic error message	
E5702	Parsing xml document failed [<error>]	Parsing of XML document failed.	See generic error message	
E5703	Validation SAXNotSupportedException [<error>]	XML parser does not support this validation method.		Inform MediData
E5704	Validation SAXNotRecognizedException [<error>]	Internal programme error		Inform MediData
E5705	Can't create instance of <name>	Specified object cannot be created		Possibly not enough memory, restart application and if problem reoccurs inform MediData.
E5801	XML VALIDATION problem [<problem>]	Problem with XML validation	Document to be sent will be rejected or not depending on xml.validation.abort.level.	
E5802	XML VALIDATION switch problem [<problem>]	Internal programme error		Inform MediData

12.4 FATAL (F0000 – F9999) log level messages

Fatal Nr	Message	Reason	Outcome	Action	Note
F0204	Current certificate not found.	Certificate for current month cannot be found	Stopped with message F2904.	Install key store with current certificates	Can only occur at application start-up.
F0205	keystorefile [<filename>] doesn't exist.	Error when initialising key store, key store file not found.	Stopped with message F2904.	Check whether key store is entered correctly in the right place with correct name and in the configuration file.	Can only occur at application start-up.
F0206	Key store initialization failed	Error when initialising key store.	Stopped with message F2904.	Check whether key store in correct place with correct name.	Can only occur at application start-up.
F0501	Property value is not a number.	Cannot process MPCommunicator configuration file.	Stopped with message F2201.	Check configuration file	Can only occur at application start-up.
F0502	The KEYSTOREPASSWORDFILE [config/_ks.mp] does not exist.	Key store password file not found.	Stopped with message F2201.	Key store password file must be available at the specified position.	
F0503	Configuration file not found.	Configuration file not found.	Application stopped.	Recreate configuration file.	

Fatal Nr	Message	Reason	Outcome	Action	Note
F0504	Decrypt of KEYSTOREPASSWORDFILE [<filename>] failed.	Key store password file could not be decrypted probably because it is corrupt.	Application stopped.	Key store password file must be made available again at the specified position. A new certificate with related password file should possibly be obtained from MediData.	
F0601	Cannot save new logging configfile.	New logging configuration file cannot be created.	No valid log configuration file exists and the application is therefore stopped.	Create a valid log configuration file.	
F0801	Get DBconnection failed.	Connection to DB failed	No communication with DB possible.	Restart application and inform MediData.	To-do: no interruption here for the moment, prog continues to run.
F0901	DB DriverClass [<drivername>] not found.	Driver for database cannot be found.	Application stopped.	Contact MediData	Can only occur at application start-up.
F1801	Initialization of XML-parser failed.	Initialisation of XML parser failed.	Application stopped.	Contain MediData	This error can only occur during development if an XML parser is used, which supports all features or if the jar file does not contain the parser.
F2201	Failed to start scheduler	Configuration error	Application stopped	See sub-message	Can only occur at application start-up.
F2202	Can't create/make needed directories	Cannot find/create required directories (send/receive/error/partner)	Application not started	Check configuration and access rights	
F2203	Don't have full access to needed directories	Application does not have full access rights (read/write/del) to required directories (send/receive/error/partner)	Application not started.	Check configuration and access rights	
F2501	Can't create InputHandler	Cannot create InputHandler	Application stopped		This error cannot occur since this exception is only defined in the interface, however will never be triggered with the FileInputHandler used.
F2904	SSL initialization failed. [SSL initialisation failed.	Application stopped immediately		Can only occur at application start-up.
F3301	Creating RemoteClientClass failed	HttpsClient instance cannot be created	Application stopped immediately	Contact MediData	This error can only occur during development if an invalid RemoteClient class has been defined in the configuration.

Fatal Nr	Message	Reason	Outcome	Action	Note
F3302	Creating RemoteClientClass failed	HttpsClient instance cannot be created	Application stopped immediately	Contact MediData	This error can only occur during development if an invalid RemoteClient class has been defined in the configuration.
F4201	F4201: Decrypting file[<filename>] failed.[<error>]	Invalid or corrupt key store password file	Application stopped immediately	Key store password file must be available again at the specified location.	Can only occur at application start-up.
F5701	Unknown XmlSaxParser-Handler		Application stopped immediately	Contact MediData	This error can only occur during development if an undefined XmlSaxParser-Handler is requested.
F5702	Creating xml parser failed.		Application stopped immediately.	Contact MediData	This error can only occur during development if an XML parser is used, which does support all features or if the jar file does not contain the parser.
F5801	XML VALIDATION problem[<problem>]	XML validation problem	Document is rejected or not depending on the xml.validation.abort.level set.	Check XML document for compliance with XML schema.	

13 Appendix C: Configuration and logging

13.1 MPC configuration files

13.1.1 Configuration file

All required information is defined in the configuration file, i.e. configuration by editing this file (ASCII text file)

The following information is given:

- EAN – Sender
- DN - Sender
- IP – Recipient
- Task / Task Manager configuration (Interval or Unix Crontabs)
- Directory paths (relative or absolute)
 - Send + Sendarchive
 - Receive, ReceiveTest
 - Error
 - Address book
- Name and path of key store (in order to facilitate the import of the new key store, the last month of the previous year is also supplied so that the certificates for the next year can be imported in December of the previous year).
- Name of send control file. All send control files must start with this name and end in .xml. This entry is case sensitive depending on the operating system.
 - SendControl

MediPort Communicator default configuration file

```
#=====
# MediPort Communicator
#-----
#
# MPCCommunicator configurationfile
#
# MPC Version      01.09.00
# Date            08.08.2007
# Author          SUM, SCA
#
# Copyright (c) 2007 MediData AG, Inc. All rights reserved.
#=====
# History:
# V01.09.00 New config parameters:
# -- 3 parameters for generic tasks
# -- CLIENT.n.SENDARCHIVE_DIR and 4 other parameters for archiving task
# -- 7 CLIENT.n.DIR.m parameters and 3 other parameters for writesendcontrol task
# V01.08.00 New config parameters:
# -- all jcifs.* properties for SMB file server access
# V01.05.00 New config parameters:
# -- CLIENT.x.y
# V01.04.00 New config parameters:
```

```
# -- PARSE_RECEIVER, RECEIVER_EANLIST, UNKNOWN_RECEIVER_SUBDIR
# -- DOCS_STATE_OUTPUTDIR, DOCS_STATE_INITIAL_DAYS
# -- mpcommunicator.getdocstate.pollinterval,
#   mpcommunicator.getdocstate.cron,
#   mpcommunicator.task.getdocstate.enable
#=====

#=====
# MINIMAL CONFIGURATION
#=====

# MediPort Communicator Sender EAN
mpcommunicator.sender.ean=2099988870017

# MediPort Communicator Sender DN
mediport.dn=uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch

# MediPort Host Name or Host Address
#   Produktiver Server: 212.243.92.201
#   Test Server       : 212.243.92.199
mpcommunicator.mediport.ip=192.168.1.243

#####
# Certificate
#####
# Keystore
keystore.name=config/EANxxxxxxxxxxxxx_mpg.keystore
keystore.passwordfile.name=config/_ks.mp

#=====
# OPTIONAL CONFIGURATION
#=====

#####
# PFADKONFIGURATION
#####

### NEUE MULTI CLIENT (SENDER) AB V01.05.00

CLIENT.1.EAN=2099988870161
CLIENT.1.SEND_DIR=data/send
CLIENT.1.SENDARCHIVE_DIR=data/send/archive
CLIENT.1.RECEIVE_DIR=data/receive
CLIENT.1.RECEIVETEST_DIR=data/receive/test
CLIENT.1.ERROR_DIR=data/error
#CLIENT.1.SEND_DIR=smb://dataserver/testuser/data/send
#CLIENT.1.SENDARCHIVE_DIR=smb://dataserver/testuser/data/send/archive
#CLIENT.1.RECEIVE_DIR=smb://dataserver/testuser/data/receive
#CLIENT.1.RECEIVETEST_DIR=smb://dataserver/testuser/data/receive/test
#CLIENT.1.ERROR_DIR=smb://dataserver/testuser/data/error
CLIENT.1.DOCSTAT_DIR=data/docstatus
CLIENT.1.PARTNER_FILE=data/partner/partnerinfo.txt
```

```
#sendcontrol-configurations for client 1
#CLIENT.1.DIR.1=data/client1/invoices1
#CLIENT.1.DIR.1.DOCATTR=Tiers_Payant
#CLIENT.1.DIR.1.DOCPRINTED=true
#CLIENT.1.DIR.1.DISTTYPE=0
#CLIENT.1.DIR.1.PRINTLANGUAGE=D
#CLIENT.1.DIR.1.TRUSTCENTEREAN=1234567890123
#CLIENT.1.DIR.1.ISPAPERINVOICE=false
#CLIENT.1.DIR.2=data/client1/invoices2
#CLIENT.1.DIR.2.DOCATTR=Tiers_Garant_Manuell

#CLIENT.2.EAN=2099988870017
#CLIENT.2.SEND_DIR=data/send
#CLIENT.2.SENDARCHIVE_DIR=data/send/archive
#CLIENT.2.RECEIVE_DIR=data/receive
#CLIENT.2.RECEIVETEST_DIR=data/receive/test
#CLIENT.2.ERROR_DIR=data/error
#CLIENT.2.DOCSTAT_DIR=data/docstatus
#CLIENT.2.PARTNER_FILE=data/partner/partnerinfo.txt

#####
# DOCS_STATE
#####

# get all the docs which are done since (today - DOCS_STATE_INITIAL_DAYS) or
# since the last time the process was started
DOCS_STATE_INITIAL_DAYS=100
DOCS_STATE_NBR_OD_ENTRIES=1000

#####
# SENDCONTROL
#####

# MediPort Communicator Sender Steuerdatei (SendControl010_xyz.xml)
# REMARK: !!! Case sensitive!!! SendControl010_ != sendcontrol010_
mpcommunicator.sender.controlfile=SendControl

#####
# TASKS (SIMPLE)
#####
# REMARK: Minimal values for the intervals are:
#   - PartnerUpdate   : 60 [minutes]
#   - Send             : 5 [seconds]
#   - Receive         : 10 [minutes]
#   - Getdocstate      : 10 [hours]
#   - Archiveddelete:
#     - delete task   : 1 [hour]
#     - store duration : 1 [day]
#   - Writesendcontrol : 1 [minute]
#####

# MediPort Communicator Partnerverzeichnis update interval[m]
mpcommunicator.partner.updateinterval=240
```

```
# MediPort Communicator send poll interval[s]
mpcommunicator.send.pollinterval=200

# MediPort Communicator receive poll interval[m]
mpcommunicator.receive.pollinterval=60

# MediPort Communicator getdocstate poll interval[h]
mpcommunicator.getdocstate.pollinterval=10

# MediPort Communicator delete sent and archived xmls poll interval[h]
mpcommunicator.archiveddelete.pollinterval=48
# MediPort Communicator send archive store duration of sent files before automatic deletion[days]
mpcommunicator.archiveddelete.maxtime=365

# MediPort Communicator write sendcontrol-xmls using configurations in CLIENT.x.DIR.y poll interval[m]
mpcommunicator.writesendcontrol.pollinterval=5

#####
# MEDIPORT SERVER
#####

# MediPort Port
mpcommunicator.mediport.port=443

#####
# Proxy Server
#####
# REMARK: Only "Basic" Proxy Authentification is supported
#####

# Use Proxy Server (true/false)
proxy.use=false

# Proxy Server Host Name or Host Address (192.168.1.122/192.168.3.51)
proxy.ip=1.1.1.1

# Proxy Server Port(8443/8100)
proxy.port=8443

# Use Proxy Server User/Password authentication (true/false)
proxy.auth.use=true

# Proxy Server authentication username
proxy.auth.username=name

# Proxy Server authentication password
proxy.auth.password=password

#####
# RMI Server
#####
# REMARK: If rmi.enable is set to true then all the
# tasks are set to false and you can only execute
```

```
# a task by calling its rmi interface.
#####
rmi.serverip=localhost
rmi.serverport=4445
rmi.servicename=MPCCommunicator-Server
rmi.enable=false

#=====
# ADVANCED CONFIGURATION
#=====

#####
# TASKS (CRON)
#####
# Cron Expressions:
# Seconds Minutes Hours Day-of-Month Month Day-of-Week
#
# Seconds(0 to 59)
# Minutes(0 to 59)
# Hours(0 to 23)
# Day-of-Month(0-31)
# Month(0-11 or JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,
# OCT, NOV and DEC)
# Day-of-Week(1-7(1 = Sunday) or SUN, MON, TUE, WED, THU,
# FRI and SAT)
#
# ranges and/or lists : "MON-WED,SAT"
#
# Wild-cards (the '*' character) every possible value of
# this field
#
# The '/' character can be used to specify increments to
# values.('0/15'~'0,15,30,45' 3/20'~'3,23,43')
#
# The '?' means no specific value for day-of-month and
# day-of-week fields.
#
# The 'L' means last for the day-of-month and day-of-week fields
# ("6L" or "FRI L"~"the last friday of the month")
#
# Example1("10 0/5 * * * ?")every 5 minutes, at 10 seconds
# after the minute(i.e. 10:00:10 am, 10:05:10 am, etc.).
# Example2("0 30 10-13 ? * WED,FRI") at 10:30, 11:30,
# 12:30, and 13:30, on every Wednesday and Friday.
# Example3("0 0/30 8-9 * 5,20 ?") every half hour between
# the hours of 8 am and 10 am on the 5th and 20th of
# every month.
#####

# MediPort Communicator Partnerverzeichnis update cron
#mpcommunicator.partner.update.cron=0 0/1 10-13 ? * WED,FRI

# MediPort Communicator send cron
#mpcommunicator.send.cron=0 0/1 10-13 ? * WED,FRI
```

```
# MediPort Communicator receive cron
#mpcommunicator.receive.cron=0 0/1 10-13 ? * WED,FRI

# MediPort Communicator getdocstate cron
#mpcommunicator.getdocstate.cron=0 0/1 10-13 ? * WED,FRI

# MediPort Communicator delete sent and archived xmls cron
#mpcommunicator.archivedelete.cron=0 0/1 10-13 ? * WED,FRI

# MediPort Communicator write sendcontrol cron
#mpcommunicator.writesendcontrol.cron=0 0/1 10-13 ? * WED,FRI

#####
# TASKS (ENABLE)
#####
# REMARK: If you run the MPCommunicator as a windows service
#   or a unix demon then you can disable the filestop
#   and the consolestop tasks.
#   If you run the MPCommunicator as an application
#   then you should at least enable one of the two
#   stoptask or you can't stop the application.
#####

# Enable (true) or disable(false) MP Communicator Tasks
mpcommunicator.task.send.enable=true
mpcommunicator.task.receive.enable=true
mpcommunicator.task.partnerupdate.enable=true
mpcommunicator.task.filestop.enable=true
mpcommunicator.task.consolestop.enable=false
mpcommunicator.task.getdocstate.enable=false
mpcommunicator.task.archive.enable=false
mpcommunicator.task.writesendcontrol.enable=false

#####
# MediPort Server
#####

# MediPort Server Servlet definitions
mediport.servlet.getadrbookinfo=GetAdrBookInfo?Sender=
mediport.servlet.getpartner=GetPartner?Sender=
mediport.servlet.postdocument=PostDocument?
mediport.servlet.getdoclist=GetDocumentList?Sender=
mediport.servlet.getdocument=GetDocument?Sender=
mediport.servlet.setstatus=SetStatus?Sender=

#####
# n generic extension tasks
# extension jobs must be numbered consecutively starting from 1,
# and either the pollinterval or the cron property must be set
#####
# Plug-in extension tasks: if a class is specified, the task is also enabled
#extension.1.job=ch.md.mp.mpc.ext.exampleclient.myjob
# Extension poll interval [minutes]
#extension.1.pollinterval=20
```



```
# Extension cron
#extension.1.cron=0 0/1 10-13 ? * WED,FRI

#extension.2.job=...
#extension.2.pollinterval=... (in minutes)
#extension.2.cron=...

#####
# SMB file server access for send, error and receive dirs.,
# i.e. Microsoft Windows "share" dirs.
#####
# For details, cf. http://jcifs.samba.org/src/docs/api/ for list of all 50+ jcifs properties that may be set
# The default username used if not specified in an SMB URL:
#jcifs.smb.client.username=auser
# The default password used if not specified in an SMB URL:
#jcifs.smb.client.password=apassword
# The default authentication domain used if not specified in an SMB URL:
#jcifs.smb.client.domain
# The IP address of the WINS server. This is only required when accessing hosts on different subnets although
it is recommended if a WINS server is provided.
#jcifs.netbios.wins
# The local network's broadcast address. It may be necessary to set this for certain network configurations
because the default of 255.255.255.255 may otherwise throw a "Network is unreachable" IOException. For
example if the local host's IP address is 192.168.1.15, the broadcast address would likely be 192.168.1.255.
#jcifs.netbios.baddr
# This is extremely rare but NetBIOS provides for a "scope id" to be used in an attempt to conceal groups of
machines on the same network. Ask your network administrator if scope id is used. If so, it must be set using
this property or name queries will fail.
#jcifs.netbios.scope
# The IP address of the local interface the client should bind to if it is different from the default. For example if
the client is to be used over a dial-up connection the IP address of the PPP interface may need to be specified
with this property.
#jcifs.smb.client.laddr
# The IP address of the local interface the client should bind to for name queries if it is different from the default.
#jcifs.netbios.laddr
# The path to an lmhosts file containing a map of IP addresses to hostnames. The format of this file is identical
to that of the Windows lmhosts file format.
#jcifs.netbios.lmhosts
```

13.1.2 Windows service configuration file

If the application is configured as a service the Java service wrapper configuration file is required.

Java service wrapper configuration file

```
#####
# Wrapper parameters for MP Communicator
#####
# Java Application
wrapper.java.command=./jre/bin/java

# Java Main class
wrapper.java.mainclass=com.medidata.mediport.communicator.service.CommunicatorServiceWrapper
```

```
# Java Classpath (include wrapper.jar) Add class path elements as
# needed starting from 1
wrapper.java.classpath.1=./lib/MPCCommunicator.jar
wrapper.java.classpath.2=./lib/iaik_jce_full_signed.jar
wrapper.java.classpath.3=./lib/MPCCommunicator_libs.jar

# Java Library Path (location of Wrapper.DLL or libwrapper.so)
wrapper.java.library.path.1=./lib

# Java Additional Parameters
#wrapper.java.additional.1=

# Initial Java Heap Size (in MB)
wrapper.java.initmemory=3

# Maximum Java Heap Size (in MB)
wrapper.java.maxmemory=128

# Application parameters. Add parameters as needed starting from 1
wrapper.app.parameter.1=./

# Port which the native wrapper code will attempt to connect to
wrapper.port=1777

#####
# Wrapper Logging parameters
#####
# Format of output for the console. (See docs for formats)
wrapper.console.format=PM

# Log Level for console output. (NONE,STATUS,DEBUG,INFO,ERROR,FATAL)
wrapper.console.loglevel=NONE

# Log file to use for wrapper output logging.
wrapper.logfile=./data/log/wrapper.log

# Format of output for the log file. (See docs for formats)
wrapper.logfile.format=LPTM

# Log Level for log file output. (NONE,STATUS,DEBUG,INFO,ERROR,FATAL)
wrapper.logfile.loglevel=STATUS

# Maximum size that the log file will be allowed to grow to before
# the log is rolled. Size is specified in bytes. The default value
# of 0, disables log rolling. May abbreviate with the 'k' (kb) or
# 'm' (mb) suffix. For example: 10m = 10 megabytes.
wrapper.logfile.maxsize=1m

# Maximum number of rolled log files which will be allowed before old
# files are deleted. The default value of 0 implies no limit.
wrapper.logfile.maxfiles=1

# Log Level for sys/event log output. (See docs for log levels)
wrapper.syslog.loglevel=NONE
```

```

#####
#
# Advanced properties
#####
# Number of seconds to allow between the time that the JVM reports
# that it is stopped and the time that the JVM process actually
# terminates. 0 means never time out. Defaults to 5 seconds.
wrapper.jvm_exit.timeout=180

# Number of seconds to allow between the time that the Wrapper
# asks the JVM to shutdown and the time that the JVM side of the
# Wrapper responds that it is stopping. 0 means never time out.
# Defaults to 30 seconds.
wrapper.shutdown.timeout=180

#####
# Wrapper NT Service parameters READ THE !!!WARNING!!!
#####
# !!!WARNING!!! - Do not modify any of these parameters when an application
# using this configuration file has been installed as a service.
# Please uninstall the service before modifying this section. The
# service can then be reinstalled.

# Name of the service
wrapper.ntservice.name=MediData MediPort Communicator

# Display name of the service
wrapper.ntservice.displayname=MediData MediPort Communicator

# Description of the service
wrapper.ntservice.description=MediData MediPort Communicator

# Service dependencies. Add dependencies as needed starting from 1
wrapper.ntservice.dependency.1=

# Mode in which the service is installed. AUTO_START or DEMAND_START
wrapper.ntservice.starttype=AUTO_START

# Priority at which the service is run. NORMAL, LOW, HIGH, or
# REALTIME
wrapper.ntservice.process_priority=NORMAL

# Allow the service to interact with the desktop.
wrapper.ntservice.interactive=false
# !!!WARNING!!! - Do not modify any of these parameters when an application
# using this configuration file has been installed as a service.
# Please uninstall the service before modifying this section. The
# service can then be reinstalled.

```

13.2 MP Communicator logging

13.2.1 Log configuration file

MediPort Communicator default log configuration file

```
##### DEFINE LOGGERS
log4j.rootCategory=INFO, A1, A2

##### FILE APPENDER A1
log4j.appender.A1=org.apache.log4j.RollingFileAppender
log4j.appender.A1.File=data/log/mpcommunicator.log
log4j.appender.A1.MaxFileSize=5000KB
log4j.appender.A1.MaxBackupIndex=3
log4j.appender.A1.layout=org.apache.log4j.PatternLayout
log4j.appender.A1.layout.ConversionPattern=%d{ISO8601} %-1x %-5p - %m%n

##### CONSOLE APPENDER A2
log4j.appender.A2=org.apache.log4j.ConsoleAppender
log4j.appender.A2.layout=org.apache.log4j.PatternLayout
log4j.appender.A2.layout.ConversionPattern=%d{ISO8601} %-1x %-5p - %m%n

##### Mail / SMTP APPENDER SMTP
log4j.appender.SMTP=org.apache.log4j.net.SMTPAppender
log4j.appender.SMTP.Threshold=ERROR
log4j.appender.SMTP.BufferSize=1
log4j.appender.SMTP.To=markus.supziger@medidata.ch,med_sum@web.de
log4j.appender.SMTP.From=mail@mail.ch
log4j.appender.SMTP.SMTPHost=slox.medidata.ch
log4j.appender.SMTP.Subject=MPCCommunicator Log Message
log4j.appender.SMTP.layout=org.apache.log4j.PatternLayout
log4j.appender.SMTP.layout.ConversionPattern=%d{ISO8601} %-1x %-5p - %m%n

##### SNMP APPENDER
log4j.appender.SNMP=org.apache.log4j.ext.SNMPTrapAppender
# ImplementationClassName: Set the value of the concrete class that implements
# the SnmpTrapSenderFacade interface.
log4j.appender.SNMP.ImplementationClassName=org.apache.log4j.ext.JoeSNMPTrapSender
log4j.appender.SNMP.layout=org.apache.log4j.PatternLayout
log4j.appender.SNMP.layout.ConversionPattern=%d %-5p - %m%n
# ManagementHost: Set the IP address of the remote host that traps should be
# sent to.
log4j.appender.SNMP.ManagementHost=pcnsum
# ManagementHostTrapListenPort: Set the port used on the remote host to listen
# for SNMP traps. The standard is 162.
log4j.appender.SNMP.ManagementHostTrapListenPort=162
# EnterpriseOID: Set the enterprise OID that will be sent in the SNMP PDU.
log4j.appender.SNMP.EnterpriseOID=1.3.6.1.4.1.24.0
# LocalIPAddress: Set the IP address of the host that is using this appender
# to send SNMP traps. This address will be encoded in the SNMP PDU, and used to
# provide things like the "agent"s IP address.
log4j.appender.SNMP.LocalIPAddress=192.168.1.99
# LocalTrapSendPort: Set the value of the port that will be used to send traps
# out from the local host.
log4j.appender.SNMP.LocalTrapSendPort=163
# GenericTrapType: Set the generic trap type for this SNMP PDU. The allowed
# values for this attribute are a part of the SNMP standard:
# 0 -- cold start
# 1 -- warm start
```

```
# 2 -- link down
# 3 -- link up
# 4 -- authentication failure
# 5 -- EGP neighbor loss
# 6 -- enterprise specific
log4j.appender.SNMP.GenericTrapType=6
# SpecificTrapType: Set the specific trap type for this SNMP PDU. Can be used
# for application and/or enterprise specific values. any value within the range
# defined for an INTEGER in the ASN.1/BER notation; i.e. -128 to 127
log4j.appender.SNMP.SpecificTrapType=0
# ApplicationTrapOID: Set the trap OID that will be sent in the SNMP PDU for
# this app.
log4j.appender.SNMP.ApplicationTrapOID=1.3.6.1.4.1.24.2
# CommunityString: Set the community string set for the SNMP session this
# appender will use. The community string is used by SNMP (prior to v.3) as a
# sort of plain-text password.
log4j.appender.SNMP.CommunityString=public
# Set the value of the system up time that will be used for the SNMP PDU.
log4j.appender.SNMP.SysUpTime=1000
```

13.2.2 MPC log file

Below is an example of an MP Communicator log file. It should be noted that no word wrapping occurs in the original log file and consequently each line always begins with the date entry.

MPCCommunicator log file

```
2003-03-21 16:09:25,187 M INFO - I2201: *****
2003-03-21 16:09:25,187 M INFO - I2201: *** Start MediPort Communicator ***
2003-03-21 16:09:25,187 M INFO - I2201: ***
2003-03-21 16:09:25,187 M INFO - I2201: *** to stop:
2003-03-21 16:09:25,187 M INFO - I2201: *** exit<ret> or Put mpcommunicator_stop file ***
2003-03-21 16:09:25,234 M INFO - I2201: *** into data directory. ***
2003-03-21 16:09:25,234 M INFO - I2201: *****
2003-03-21 16:09:25,234 M INFO - I2201: Starting MPCCommunicator V1.0.0
2003-03-21 16:09:25,453 M INFO - I2204: ----- Scheduler schedule jobs -----
2003-03-21 16:09:25,515 M INFO - I2205: grp_mg.job_PartnerUpdate will run at[Fri Mar 21 16:09:30 CET 2003] & repeat[-1/7200000]
2003-03-21 16:09:25,531 M INFO - I2205: grp_mg.job_Receive will run at[Fri Mar 21 16:09:30 CET 2003] & repeat[-1/60000]
2003-03-21 16:09:25,531 M INFO - I2205: grp_mg.job_SendControlFile will run at[Fri Mar 21 16:09:30 CET 2003] & repeat[-1/10000]
2003-03-21 16:09:25,531 M INFO - I2205: grp_mg.job_ConsoleStop will run at[Fri Mar 21 16:09:30 CET 2003] & repeat[-1/10000000000]
2003-03-21 16:09:25,531 M INFO - I2206: ----- Scheduler started -----
2003-03-21 16:09:30,625 P INFO - I2904: Current cert = 03-2003
2003-03-21 16:09:31,718 S INFO - I1001: Start processing new SendControlFile[NbrOfDocs[8] DocsProcessed[0] SendfileSize[1883] LastModified[Thu Feb 27 16:16:07 CET 2003]]
2003-03-21 16:09:37,593 P INFO - I2902: ADRBOOK UNCHANGED.
2003-03-21 16:09:37,625 P INFO - I0801: New partnerfile[.data\partner\partnerinfo.txt] created.
2003-03-21 16:09:55,734 S INFO - I2501: SEND DOC OK FILENAME[.data\send\HOi_0001.xml]
RECEIVERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[100]
MDID[5336] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSD HospitalInvoiceRequest_300.xsd] JOB[1048259375312]
2003-03-21 16:09:56,640 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259396187TMP0.XML]
```

SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5323] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259370109]
2003-03-21 16:10:13,921 S INFO - I2501: SEND DOC OK FILENAME[.data\send\Hoi_0001a.xml]
RECEIVERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[100]
MDID[5337] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSD HospitalInvoiceRequest_300.xsd] JOB[1048259375312]
2003-03-21 16:10:14,953 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259414484TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5324] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259370109]
2003-03-21 16:10:16,859 S INFO - I2501: SEND DOC OK FILENAME[.data\send\Hoi_0002.xml]
RECEIVERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[100]
MDID[5338] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSD HospitalInvoiceRequest_300.xsd] JOB[1048259375312]
2003-03-21 16:10:17,703 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259417218TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5325] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259370109]
2003-03-21 16:10:20,390 S INFO - I2501: SEND DOC OK FILENAME[.data\send\Hoi_0002a.xml]
RECEIVERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[100]
MDID[5339] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSD HospitalInvoiceRequest_300.xsd] JOB[1048259375312]
2003-03-21 16:10:21,437 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259420968TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5326] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259370109]
2003-03-21 16:10:23,203 S INFO - I2501: SEND DOC OK FILENAME[.data\send\Hoi_0003.xml]
RECEIVERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[100]
MDID[5340] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSD HospitalInvoiceRequest_300.xsd] JOB[1048259375312]
2003-03-21 16:10:24,125 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259423656TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5327] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259370109]
2003-03-21 16:10:26,265 S INFO - I2501: SEND DOC OK FILENAME[.data\send\Hoi_0003a.xml]
RECEIVERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[100]
MDID[5341] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSD HospitalInvoiceRequest_300.xsd] JOB[1048259375312]
2003-03-21 16:10:27,046 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259426625TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5328] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259370109]
2003-03-21 16:10:28,875 S INFO - I2501: SEND DOC OK FILENAME[.data\send\Horem_0001.xml]
RECEIVERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[100]
MDID[5342] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSD InvoiceReminderRequest_300.xsd] JOB[1048259375312]
2003-03-21 16:10:29,734 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259429234TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5329] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259370109]
2003-03-21 16:10:30,875 S INFO - I2501: SEND DOC OK FILENAME[.data\send\Horem_0006.xml]
RECEIVERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[100]
MDID[5343] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSD InvoiceReminderRequest_300.xsd] JOB[1048259375312]
2003-03-21 16:10:31,734 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259431250TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5330] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259370109]
2003-03-21 16:10:32,562 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259432093TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5331] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259370109]
2003-03-21 16:10:33,468 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRREX03001048259433031TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5332] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDInvoiceReminderRequest_300.xsd] JOB[1048259370109]
2003-03-21 16:10:34,375 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRREX03001048259433906TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5333] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDInvoiceReminderRequest_300.xsd] JOB[1048259370109]


```

2003-03-21 16:10:35,218 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259434734TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5334] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259370109]
2003-03-21 16:10:36,046 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259435578TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5335] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259370109]
2003-03-21 16:11:31,578 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259491125TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5336] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259490109]
2003-03-21 16:11:32,625 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259492125TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5337] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259490109]
2003-03-21 16:11:33,421 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259492953TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5338] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259490109]
2003-03-21 16:11:34,281 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259493781TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5339] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259490109]
2003-03-21 16:11:35,109 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259494625TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5340] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259490109]
2003-03-21 16:11:35,906 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRHSP03001048259495437TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5341] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259490109]
2003-03-21 16:11:36,828 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRREX03001048259496375TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5342] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDInvoiceReminderRequest_300.xsd] JOB[1048259490109]
2003-03-21 16:11:37,687 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRREX03001048259497250TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5343] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDInvoiceReminderRequest_300.xsd] JOB[1048259490109]
2003-03-21 16:12:30,578 R INFO - I2903: NO NEW DOCUMENTS ON SERVER.
2003-03-21 16:13:20,171 M INFO - I2207: ----- Shutting Down -----
2003-03-21 16:13:20,171 M INFO - I2203: RunningSince[Fri Mar 21 16:09:25 CET 2003] NbrOfJobsExecuted[29] NbrDocs
Sent[8] NotSent[0] Received[21]
2003-03-21 16:13:20,171 M INFO - I2208: ----- Shutdown Complete -----
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5341] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDHospitalInvoiceRequest_300.xsd] JOB[1048259490109]
2003-03-21 16:11:36,828 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRREX03001048259496375TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5342] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDInvoiceReminderRequest_300.xsd] JOB[1048259490109]
2003-03-21 16:11:37,687 R INFO - I2401: RECEIVE DOC OK
FILENAME[.data\receive\test\IRREX03001048259497250TMP0.XML]
SENDERDN[uid=EAN2099988870017,ou=Test,ou=MPGateway,ou=Kostentraeger,o=medidata.ch] STATETYPEID[200]
MDID[5343] DOCTYPE[http://www.xmlData.ch/xmlInvoice/XSDInvoiceReminderRequest_300.xsd] JOB[1048259490109]
2003-03-21 16:12:30,578 R INFO - I2903: NO NEW DOCUMENTS ON SERVER.
2003-03-21 16:13:20,171 M INFO - I2207: ----- Shutting Down -----
2003-03-21 16:13:20,171 M INFO - I2203: RunningSince[Fri Mar 21 16:09:25 CET 2003] NbrOfJobsExecuted[29] NbrDocs
Sent[8] NotSent[0] Received[21]
2003-03-21 16:13:20,171 M INFO - I2208: ----- Shutdown Complete -----

```

14 Appendix D: Scheduler configuration

14.1 Cron Expressions

Cron-Expressions are used to configure instances of CronTrigger. Cron-Expressions are strings that are actually made up of six sub-expressions, that describe individual details of the schedule. These sub-expression are separated with white-space, and represent:

Seconds
Minutes
Hours
Day-of-Month
Month
Day-of-Week

An example of a complete cron-expression is the string "0 0 12 ? * WED" - which means "every Wednesday at 12:00 pm".

Individual sub-expressions can contain ranges and/or lists. For example, the day of week field in the previous (which reads "WED") example could be replaced with "MON-FRI", "MON, WED, FRI", or even "MON-WED,SAT".

Wild-cards (the '*' character) can be used to say "every" possible value of this field. Therefore the '*' character in the "Month" field of the previous example simply means "every month". A '*' in the Day-Of-Week field would obviously mean "every day of the week".

All of the fields have a set of valid values that can be specified. These values should be fairly obvious - such as the numbers 0 to 59 for seconds and minutes, and the values 0 to 23 for hours. Day-of-Month can be any value 0-31, but you need to be careful about how many days are in a given month! Months can be specified as values between 0 and 11, or by using the strings JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV and DEC. Days-of-Week can be specified as values between 1 and 7 (1 = Sunday) or by using the strings SUN, MON, TUE, WED, THU, FRI and SAT.

The '/' character can be used to specify increments to values. For example, if you put '0/15' in the Minutes field, it means 'every 15 minutes, starting at minute zero'. If you used '3/20' in the Minutes field, it would mean 'every 20 minutes during the hour, starting at minute three' - or in other words it is the same as specifying '3,23,43' in the Minutes field.

The '?' character is allowed for the day-of-month and day-of-week fields. It is used to specify "no specific value". This is useful when you need to specify something in one of the two fields, but not the other. See the examples below (and CronTrigger JavaDOC) for clarification.

The 'L' character is allowed for the day-of-month and day-of-week fields. This character is short-hand for "last", but it has different meaning in each of the two fields. For example, the value "L" in the day-of-month field means "the last day of the month" - day 31 for January, day 28 for February on non-leap years. If used in the day-of-week field by itself, it simply means "7" or "SAT". But if used in the day-of-week field after another value, it means "the last xxx day of the month" - for example "6L" or "FRIL" both mean "the last friday of the month". When using the 'L' option, it is important not to specify lists, or ranges of values, as you'll get confusing results.

Here are a few more examples of expressions and their meanings - you can find even more in the JavaDOC for CronTrigger

CronTrigger Example 1 - an expression to create a trigger that simply fires every 5 minutes

```
"0 0/5 * * * ?"
```

CronTrigger Example 2 - an expression to create a trigger that fires every 5 minutes, at 10 seconds after the minute (i.e. 10:00:10 am, 10:05:10 am, etc.).

```
"10 0/5 * * * ?"
```

CronTrigger Example 3 - an expression to create a trigger that fires at 10:30, 11:30, 12:30, and 13:30, on every Wednesday and Friday.

```
"0 30 10-13 ? * WED,FRI"
```

CronTrigger Example 4 - an expression to create a trigger that fires every half hour between the hours of 8 am and 10 am on the 5th and 20th of every month. Note that the trigger will NOT fire at 10:00 am, just at 8:00, 8:30, 9:00 and 9:30

```
"0 0/30 8-9 * 5,20 ?"
```

15 Appendix E: Release Notes

MediPort Communicator Release Notes

*** Description ***

MediPort Communicator is a 100% java server application which is able to transport standard xml invoices via MediPort automatically. It uses a java scheduler to execute the tasks (Send/Receive/PartnerUpdate/GetDocState etc.).

*** New Release 23-03-2012 V01.13.00 ***

New features

- support for new xml schemas added (send and receive):
 - * ekarus_x230.xsd
 - * ekarus_x240.xsd
 - * ekarus_x250.xsd
- received xml 4.3 and ekarus documents now only contain 1 via element with intermediate EAN of MediData

Following known bugs fixed

- none

*** New Release 30-01-2012 V01.12.00 ***

New features

- support for new xml schemas added (send and receive):
 - * generalCreditRequest_430.xsd
 - * generalCreditResponse_430.xsd
 - * generalFormRequest_430.xsd
 - * generalFormResponse_430.xsd
 - * generalNotification_430.xsd
 - * xmitDocuments_410.xsd
- schemaID for xmitDocuments_410.xsd added (send control and generation of send control)
- generation of send control by plugins possible
- configurable xslt transformation of received documents

Following known bugs fixed

- none

*** New Release 29-11-2011 V01.11.00 ***

New features

- EBPP document types added: EbppInvoiceRequest_100.xsd, EbppAdminMessage_100.xsd
- send control file generation changed: now document types requiring DocAttr="direct" can only use "direct", regardless of configuration. For all other documents, if nothing is configured and workflow cannot be determined from xml then "direct" is always used

Following known bugs fixed

- XML 4.3 schemas fixed: validation previously required internet connection, now not anymore
- send control file generation: DocAttr="direct" was not possible until now

*** New Release 24-03-2011 V01.10.00 ***

New features

- support for new xml 4.3 schemas added (send and receive):
 - * generalContainer_430.xsd
 - * generalInvoiceRequest_430.xsd
 - * generalInvoiceResponse_430.xsd
 - * hospitalMCDRequest_430.xsd
 - * hospitalMCDResponse_430.xsd
 - * statusRequest_430.xsd
 - * statusResponse_430.xsd
- write send control-task:if workflow of an xml cannot be determined, the xml is moved to the client's error directory instead of only being ignored

Following known bugs fixed

- none

*** New Release 17-01-2011 V01.09.03 ***

Following known bugs fixed

- compatibility with Mac OS X restored

*** New Release 10-08-2010 V01.09.02 ***

New features

- new jre 1.6.0_15
- Solaris x86 version added, Solaris Sparc version removed
- Windows version is 32-bit, runs on Windows from Windows 2000 up to Windows 7
- full Windows 7 and Vista compatibility: at installation, a separate data-directory (not write-protected for standard users) for all user-configurable and -written files can be specified
- added log-entry for the most common configuration errors
- send control file generation task: xml-filename-suffixes ".XML" and ".Xml" are changed to ".xml"
- send control file generation task: incomplete xmls are ignored (e.g. concurrent read & write of the same file)
- xmls to be sent are schema-validated
- multiple sending of identical documents is prevented up to certain limits
- received xmls: the intermediary EAN is changed to 7601001304307 if it was different
- renamed all commands for Windows-versions, so all versions now have underscores "_"

Following known bugs fixed

- in scheduler mode, mpcommunicator_stop file existing at startup is removed instead of causing immediate shutdown
- document status tracking: server is now queried repeatedly until no new states are reported (prev. querying was stopped after less than a requested max. number of states was returned)

*** New Release 22-08-2008 V01.09.01 ***

- compatibility for Windows Vista added (new installer generated; no other changes)

*** New Release 10-08-2007 V01.09.00 ***

New features

- any number of user-written "plugin" tasks may be added (e.g. to prepare xml invoices)
- no send control xml files are needed anymore - instead, for every client multiple sets of send parameters may be individually configured
- all sent xmls may be archived, including their send control files, and these archives can automatically be deleted after a specified number of days
- if mpcommunicator.mediport.ip is changed, the partners are automatically changed appropriately after the next restart of the Communicator
- clients may now share the same directories, i.e. xmls with different EAN numbers may be sent from the same send directory
- keystore-file can now be copied from the certificate CD to the config dir. without renaming
- the shutdown-reason is shown in the log
- xmls with a different intermediary EAN than 7601001304307 are not sent
- trailing spaces in path configurations are trimmed
- a few modifications in SendControl10.xsd, e.g. FileName lengths now up to 256 characters
- a missing schema name now only discards respective xml instead of all xmls in current send control
- the parameter CLIENTS.MAXNUMBER is not used anymore, instead clients may be numbered using any positive integers
- various improved log messages

Following known bugs fixed

- a document status 2 was previously erroneously returned as 4
- previously the status of only 8 x DOCS_STATE_NBR_OD_ENTRIES documents could be retrieved
- the correct namespace "xmitDocuments" was added for xmitDocuments_400.xsd documents

*** New Release 23-01-2006 V01.08.00 ***

New features

- SMB file server access for send, error and receive directories, i.e. Microsoft Windows "share" directories
- if a document is not sent, the send control file containing the entry for that document is also moved to the error directory (in addition to the document itself, if found). Existing send control files in the error directory are not overwritten, so the newly-copied one may be assigned a new unique filename not used since the last startup of the MPC.

Following known bugs fixed

- SendControl10.xsd not deleted anymore from data/send/ directory when TestSuite is executed

- all *.exe and *.bat files now installed on all Windows platforms (previously not installed on Windows Server 2003)

*** New Release 31-10-2005 V01.07.00 ***

New features

- partner update (address book): upon execution of the partner update task, only active partners remain in the internal database and only active partners are written to the partnerinfo.txt file. Documents sent to another (e.g. inactive) partner will therefore be sent to the printcenter.
- support for new document types added (send and receive):
 - * generalinvoicerequest_410.xsd
 - * generalinvoiceresponse_410.xsd
 - * generalcreditrequest_400.xsd
 - * generalcreditresponse_400.xsd
 - * hospitalcreditrequest_400.xsd
 - * hospitalcreditresponse_400.xsd
 - * xmitdocuments_400.xsd
- multiclient now supports any number of clients, configurable with the CLIENTS.MAXNUMBER property
- support for new parameters TrustCenterEAN and IsPaperInvoice in sendcontrol file and in PostDocument servlet call, DocSize now also sent as parameter to PostDocument servlet.
- JVM 1.4.2-8 included in release of all platforms
- install log created during installation

Following known bugs fixed

- order of JUnit tests changed, so that partners are updated before being referenced by subsequent tests
- org.apache.regexp.RE replaced by java.util.regex.Pattern and java.util.regex.Matcher, as RE can cause StackOverflowErrors
- installer now queries desired install path in console mode installation

*** New Release 26-09-2005 V01.06.02 ***

New features

- multiclient supports now up to 100 clients

Following known bugs fixed

- none

*** New Release 10-08-2005 V01.06.01 ***

New features

- none

Following known bugs fixed

- DocState -> DocStatus changed in GetStateJob. The XML should
now be like it is defined in XML-Schema MPCGetDocumentList.xsd

*** New Release 01-07-2005 V01.06.00 ***

New features

- The MPC supports the new DocAttr Tiers_Garant_Direct. This is used for
responding to a TG request by sending back the corresponding TG Dokument.
Only works with MPSTServer V02.xx.xx or higher
- New DB: Added support for the new TG Request Document Typ (Pull Standard)

Following known bugs fixed

- When the GetDocStat returns an empty response then the MPC looped through all
the 999 iterations. this is now corrected and he stops.
- Correction in getDocStates: in case we get no CResponse (e.g. No auth)
- When the SendJob had a connection problem then he sometimes looped. this is
fixed now.

*** New Release 31-05-2005 V01.05.02 ***

Changes

- The MPC now stops if he fails to delete a file (doc or SendControl)

*** New Release 02-05-2005 V01.05.01 ***

Following known bugs fixed

- Synchronized access to HttpClient in PartnerUpdate.
- IOHandler.output (Dok) with MCC config.

*** New Release 28-04-2005 V01.05.00 ***

New features

- SNMP Trap Sender in Log Configuration.
- Calling of the CheckServer servlet before doing a getpartner.
- Real MultiClient Support
- All Config Path now can be in UNC notation (//pcnsum/mpc_test/daten),
absolute (D:/mpc_test/daten) or relative (./daten)

Following known bugs fixed

- PostDocumentFilter: The role parameter in PostDocument is now also set for
InvoiceResponse and ReminderResponse.

*** New Release 17-02-2005 V01.04.01 ***

Following known bugs fixed

- Multiuser support had an error in the processing of received Reminders
(GetDoc).

*** New Release 27-09-2004 V01.04.00 ***

New features

- Multiuser support. The received documents for different receivers are stored in separate directories (Invoice/Reminder : Insurance EAN, InvoiceResponse/ReminderResponse : Biller EAN)
- New Task GetDocState : returns the state of all documents sent to MediPort by the user of this MPC.
- Support for Invoice / Reminder 4.0 (changes in DB)

Changes

- Changes in Receive Job:
the getDocList is called until it returns an empty list of docs (because since MPSTServer V1.9.0 the getDocList is limited and only returns x Docs in one single request).
- W4202:certificate is going to expire soon. is disabled.
- The MPC can only be started once per Installation (always maps the configured RMI Port)

Following known bugs fixed

- There were some situations where the stored sendcontrolfile in the db was not deleted before a new entry had to be inserted. this is fixed now.

*** New Release 05.04-2004 V01.3.0 ***

New features

- Email notification in case of warn/error
- Linux installer

Following known bugs fixed

- Now the document that is sent to a insurance that is not connected to MediPort will be printed.
REQUIRES: that the full address of the insurance is in the xml.
- the communicator now sends the Role=test/production flag set in the xml document.
- Partner corrected to Partner (mpcommunicator.partner.file)
- Corrections in testsuite
- temporary Outputfile name now ends with "T" instead of "TMP" to ensure that all filenames are shorter than 32 chars.

```
*****
*** New Release 20.08-2003 V01.2.0          ***
*****
```

New features

- Support for JVM 1.4 added
- use of singed SecurityProvider added (Sun JVM 1.4)
- Installer

```
*****
*** New Release 06-2003 V01.1.0          ***
*****
```

New features

- absolute path in configuration allowed
- more then one sendcontrolfile allowed
- use of ISO identifiers for the ParintLanguage(DE,IT,FR)