National Bank of Georgia

Annex E – Participant Connectivity

Automated Transfer System Upgrade Delivery Project

Version: 1.00
Date: 2025-06-17

Table of Contents

1. Introduction

Overview

Montran Gateway is a complete solution to integrate ATS System with Participant bank's back-office and client services. It links the bank's existing internal systems with the ATS via JMS API over Virtual Private Network (VPN) for Straight-Through Processing (STP) of all payment and message traffic.

Montran Gateway communicates with the Back-Office Applications through a file-based connection or through JMS API message-oriented middleware (referred to as MQ below). These two connection types are explained in the following sections.

2. Overview of the System Interface

2.1. Prerequisites for participants

To run Montran Gateway application on the workstation, a set of steps should be followed. Please note that this configuration should be done for each workstation that needs to run the Gateway application.

The following steps are needed for the Gateway environment:

1. Java installation. Oracle Java can be used if a usage license is acquired from Oracle. As an alternative an open-source JAVA version can be used. We recommend Azul implementation of IcedTea-Web and Zulu OpenJDK build version 8.
2. Download Zulu OpenJDK version 8 as zip and extract it. The following link https://www.azul.com/downloads/?version=java-8-lts&package=jdk can be used. The extracted result can be placed anywhere on disk. Remember the path location. E.g.: c:\Java\jdk8.0.312-zulu8.58.0.13.
3. Set the JAVA_HOME and PATH variables to the location where the JDK was extracted. In Windows using Command Prompt (cmd), run the following two commands:

   ```
   setx JAVA_HOME "C:\Java\<jdk8_location>"
   setx PATH "%PATH%;%JAVA_HOME%\bin"
   ```

   Where <jdk8_location> is the location of the extracted JDK as described at point 2. E.g.: c:\Java\jdk8.0.312-zulu8.58.0.13.

   After the above commands are executed, open a new command prompt and run the command:

   ```
   java -version
   ```

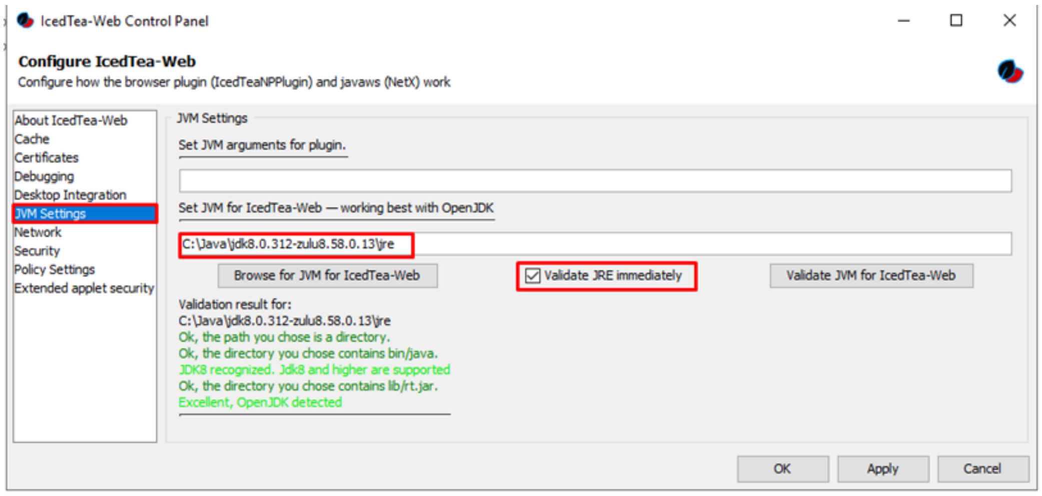
   The following output (or similar) should be displayed:

   ```
   openjdk version "1.8.0_312"
   OpenJDK Runtime Environment (Zulu 8.58.0.13-CA-win64) (build 1.8.0_312-b07)
   OpenJDK 64-Bit Server VM (Zulu 8.58.0.13-CA-win64) (build 25.312-b07, mixed mode)
   ```
4. Download Java Webstart. This software does not come with the JDK package anymore and needs to be installed manually. The product is also open-source and is called IcedTea-Web. Please follow the instructions provided here https://docs.azul.com/core/icedteaweb/installation to install IcedTea-Web.

   There will be two important applications inside it:

   - itweb-settings.exe
   - javaws.exe
5. IcedTea-Web has to be configured to use the Java JDK as follows: Open its settings by running itweb-settings.exe.

From the configuration, go to "JVM Settings" and on the second text box labeled "Set JVM for IcedTea-Web" set the path to your JAVA. E.g: C:\Java\jdk8.0.312-zulu8.58.0.13\jre.


IcedTea-Web Control Panel

Configure IcedTea-Web
Configure how the browser plugin (IcedTeaNPPlugin) and javaws (NetX) work

JVM Settings

Set JVM arguments for plugin.

Set JVM for IcedTea-Web — working best with OpenJDK: C:\Java\jdk8.0.312-zulu8.58.0.13\jre

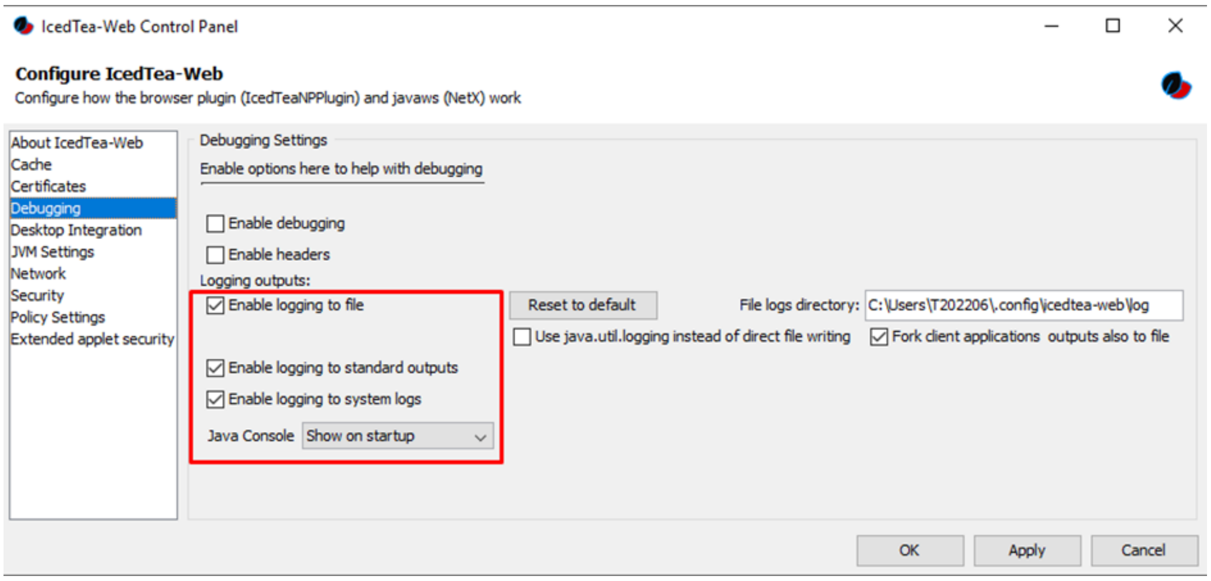
[✓] Validate JRE immediately

```
Validation result for:
C:\Java\jdk8.0.312-zulu8.58.0.13\jre
Ok, the path you chose is a directory.
Ok, the directory you chose contains bin/java.
JDK8 recognized. Jdk8 and higher are supported
Ok, the directory you chose contains lib/rt.jar.
Excellent, OpenJDK detected
```

Check that it passed the auto validation, then save the settings.

6. Next, enable Java Console logging. This helps when debugging is required.

   Go to "Debugging".


IcedTea-Web Control Panel

Configure IcedTea-Web
Configure how the browser plugin (IcedTeaNPPlugin) and javaws (NetX) work

Debugging Settings

Enable options here to help with debugging

[ ] Enable debugging
[ ] Enable headers

Logging outputs:

[✓] Enable logging to file

File logs directory: C:\Users\T202206\.config\icedtea-web\log

[ ] Use java.util.logging instead of direct file writing
[✓] Fork client applications outputs also to file
[✓] Enable logging to standard outputs
[✓] Enable logging to system logs

Java Console: Show on startup

2.2. Running the Gateway Application

Montran Gateway application uses JAVA Web Start technology to facilitate application delivery via HTTPS.

To start Montran Gateway, participants must use the ATS menu entry: **Routing -> Gateway.**

Click **Start**.

The Gateway is downloaded to the workstation (file name "Gateway(x)", extension ". jnlp").

Note: Please note that is not necessary to click Start each time Gateway requires starting. When a new session is started, after the selected Gateway configuration was complete and saved on disk (details in sub-chapter "Gateway Configuration"), open the previous saved jnlp file, click OK and follow the steps described in the second paragraph below.

Open the jnlp file.

- If Gateway is accessed for the first time:

A screen with Gateway configuration options is displayed (details in sub-chapter "Gateway Configuration"). This allows the user to specify the operation mode and the transport queues.

- If Gateway is accessed after the initial configuration:

A screen with Gateway reconfiguration options is displayed. If the initial configuration persisted on the disk is ok, click **No**.

If digital signature is enabled a dialog prompting token PIN entry is displayed. Enter token PIN

Click **OK**.

Now the Gateway session is open and the Monitoring Screen of the Gateway opens (details in the section "Gateway Monitor").

2.3. Gateway Configuration

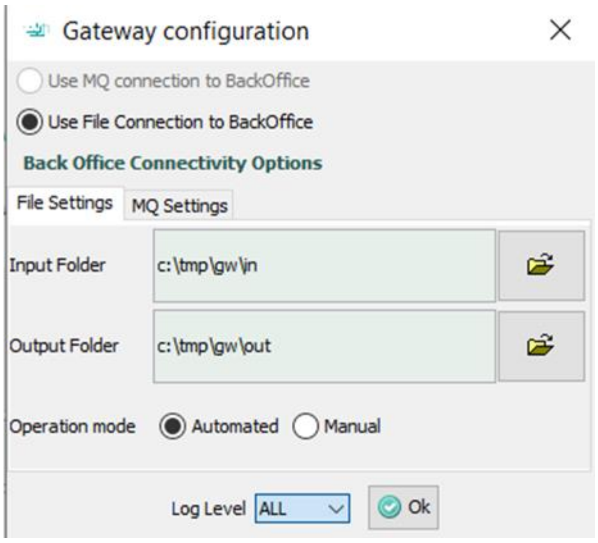
Gateway configuration allows the user to specify the operation mode and the MQ message queues or directories on Participant's Core Banking used in the STP process.

Two connectivity modes between the Gateway and the Participant Core Banking System are available.

1. **File connection**

In this mode communication is done through a shared set of input and output directories through which payment files are exchanged between Gateway and the Participant core banking.

Select the File Settings tab, and fill in the Input/Output Folder fields, using the browse button (...).


Gateway configuration

Use MQ connection to BackOffice [✓ Use File Connection to BackOffice]

Back Office Connectivity Options

File Settings | MQ Settings

Input Folder: c:\tmp\gw\in

Output Folder: c:\tmp\gw\out

Operation mode: [✓ Automated] Manual

Log Level: ALL

Note: Neither the input nor the output folder cannot be the root of a drive or located in a directory where Gateway has no access rights.

If the mode of operation is **Automated**, the files will be automatically collected by the Gateway and will be sent to the ATS.

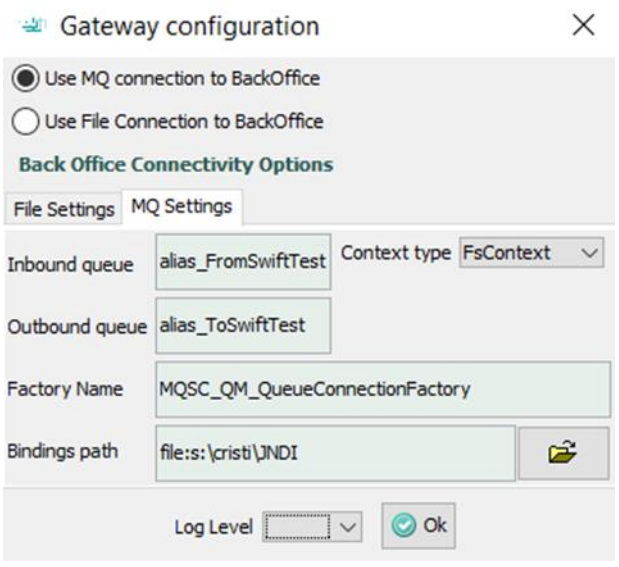
If the mode is **Manual**, no files will be automatically collected and sent by the Gateway to the ATS. In this case, the user must select the message required to be sent, by right clicking each file and choosing the option "Send".

Then click **OK**.

2. MQ connection

Select the MQ Settings tab and fill in

- the Inbound/Outbound Queue and Factory text fields
- context type that can be either
  - generic JMS API (FsContext) and it is using a .bindings path field
  - specific MQ provider such as WEBSPHERE SIB.


Gateway configuration

(●) Use MQ connection to BackOffice
( ) Use File Connection to BackOffice

Back Office Connectivity Options

File Settings | MQ Settings

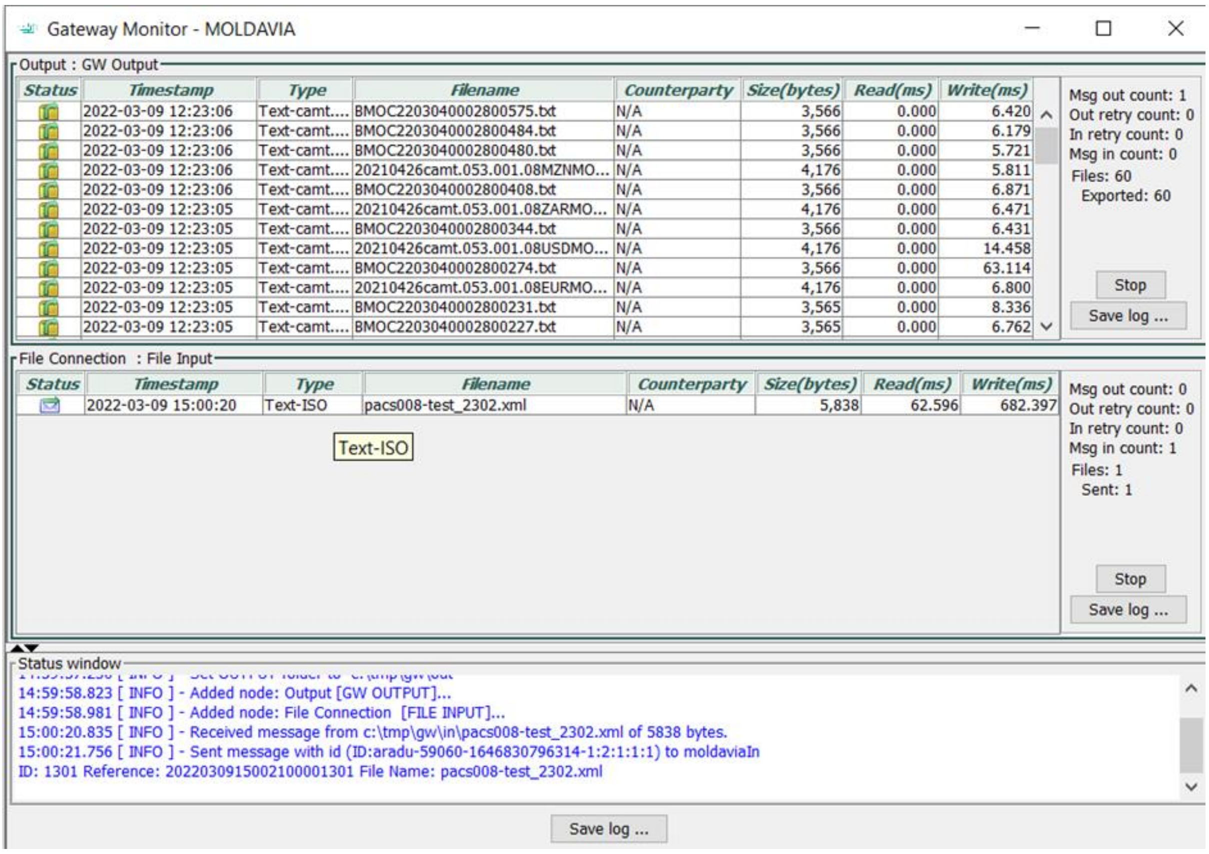
Inbound queue: alias_FromSwiftTest
Context type: FsContext
Outbound queue: alias_ToSwiftTest
Factory Name: MQSC_QM_QueueConnectionFactory
Bindings path: file:s:\cristi\JNDI

Then click **OK**.

The selected configuration is stored on disk, the next time Gateway is started the same settings will be used.

Log Level

2.4. Gateway Monitor



Gateway Monitor - MOLDAVIA

Output : GW Output

Status	Timestamp	Type	Filename	Counterparty	Size(bytes)	Read(ms)	Write(ms)
	2022-03-09 12:23:06	Text-camt....	BMOC2203040002800575.txt	N/A	3,566	0.000	6.420
	2022-03-09 12:23:06	Text-camt....	BMOC2203040002800484.txt	N/A	3,566	0.000	6.179
	2022-03-09 12:23:06	Text-camt....	BMOC2203040002800480.txt	N/A	3,566	0.000	5.721
	2022-03-09 12:23:06	Text-camt....	20210426camt.053.001.08MZNMO...	N/A	4,176	0.000	5.811
	2022-03-09 12:23:06	Text-camt....	BMOC2203040002800408.txt	N/A	3,566	0.000	6.871
	2022-03-09 12:23:05	Text-camt....	20210426camt.053.001.08ZARMO...	N/A	4,176	0.000	6.471
	2022-03-09 12:23:05	Text-camt....	BMOC2203040002800344.txt	N/A	3,566	0.000	6.431
	2022-03-09 12:23:05	Text-camt....	20210426camt.053.001.08USDMO...	N/A	4,176	0.000	14.458
	2022-03-09 12:23:05	Text-camt....	BMOC2203040002800274.txt	N/A	3,566	0.000	63.114
	2022-03-09 12:23:05	Text-camt....	20210426camt.053.001.08EURMO...	N/A	4,176	0.000	6.800
	2022-03-09 12:23:05	Text-camt....	BMOC2203040002800231.txt	N/A	3,565	0.000	8.336
	2022-03-09 12:23:05	Text-camt....	BMOC2203040002800227.txt	N/A	3,565	0.000	6.762

Msg out count: 1
Out retry count: 0
In retry count: 0
Msg in count: 0
Files: 60
Exported: 60

File Connection : File Input

Status	Timestamp	Type	Filename	Counterparty	Size(bytes)	Read(ms)	Write(ms)
	2022-03-09 15:00:20	Text-ISO	pacs008-test_2302.xml	N/A	5,838	62.596	682.397

Msg out count: 0
Out retry count: 0
In retry count: 0
Msg in count: 1
Files: 1
Sent: 1

```
14:59:58.823 [ INFO ] - Added node: Output [GW OUTPUT]...
14:59:58.981 [ INFO ] - Added node: File Connection  [FILE INPUT]...
15:00:20.835 [ INFO ] - Received message from c:\tmp\gw\in\pacs008-test_2302.xml of 5838 bytes.
15:00:21.756 [ INFO ] - Sent message with id (ID:aradu-59060-1646830796314-1:2:1:1:1) to moldaviaIn
ID: 1301 Reference: 20220309150021000001301 File Name: pacs008-test_2302.xml
```

Gateway Monitor shows statistics for input and output in both cases:

- Msg in – messages received;
- Msg out – messages sent;
- In/Out Retry count – number of retries to perform the receive/send operation in case of communication failure.

Use the **Stop** buttons to stop the file transmission.

To close the application, use the Close Window button.

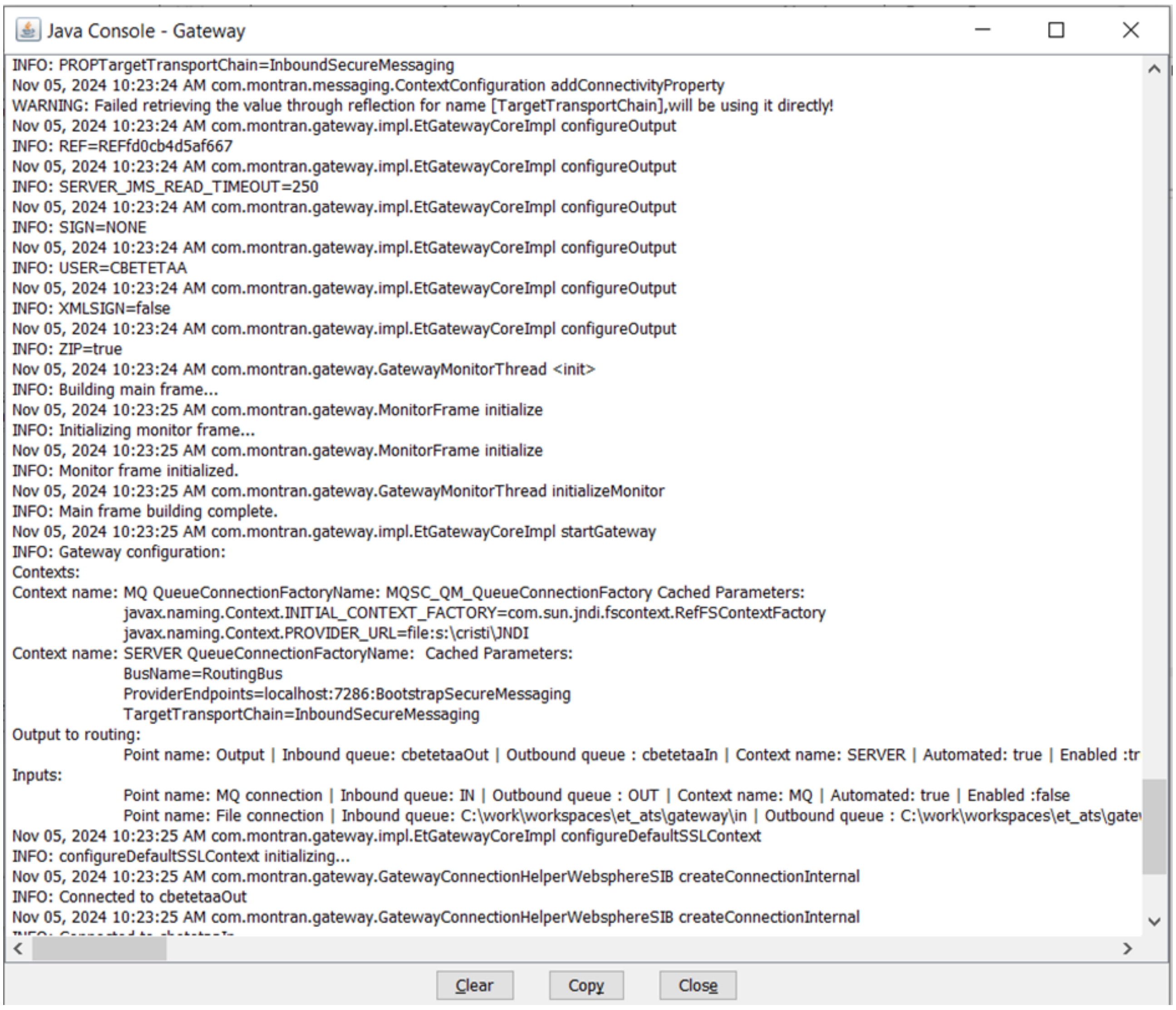
3. Technical information

3.1. Gateway files and structure

The gateway configuration is stored in the <USERPROFILE>\gateway folder. The filename is gatewayConfiguration.xml.

The gateway logs are stored in the <USERPROFILE>\gateway\log folder. The filename is either gateway.log (for the latest log) or gateway.log.<YYYY-MM-DD> (for the archived logs).

If JAVA console is enabled for the Java WebStart then at each Gateway run the console log will show up and list debugging information.



```
INFO: PROPTargetTransportChain=InboundSecureMessaging
Nov 05, 2024 10:23:24 AM com.montran.messaging.ContextConfiguration addConnectivityProperty
WARNING: Failed retrieving the value through reflection for name [TargetTransportChain],will be using it directly!
Nov 05, 2024 10:23:24 AM com.montran.gateway.impl.EtGatewayCoreImpl configureOutput
INFO: REF=REFfd0cb4d5af667
Nov 05, 2024 10:23:24 AM com.montran.gateway.impl.EtGatewayCoreImpl configureOutput
INFO: SERVER_JMS_READ_TIMEOUT=250
Nov 05, 2024 10:23:24 AM com.montran.gateway.impl.EtGatewayCoreImpl configureOutput
INFO: SIGN=NONE
Nov 05, 2024 10:23:24 AM com.montran.gateway.impl.EtGatewayCoreImpl configureOutput
INFO: USER=CBETETAA
Nov 05, 2024 10:23:24 AM com.montran.gateway.impl.EtGatewayCoreImpl configureOutput
INFO: XMLSIGN=false
Nov 05, 2024 10:23:24 AM com.montran.gateway.impl.EtGatewayCoreImpl configureOutput
INFO: ZIP=true
Nov 05, 2024 10:23:24 AM com.montran.gateway.GatewayMonitorThread <init>
INFO: Building main frame...
Nov 05, 2024 10:23:25 AM com.montran.gateway.MonitorFrame initialize
INFO: Initializing monitor frame...
Nov 05, 2024 10:23:25 AM com.montran.gateway.MonitorFrame initialize
INFO: Monitor frame initialized.
Nov 05, 2024 10:23:25 AM com.montran.gateway.GatewayMonitorThread initializeMonitor
INFO: Main frame building complete.
Nov 05, 2024 10:23:25 AM com.montran.gateway.impl.EtGatewayCoreImpl startGateway
INFO: Gateway configuration:
Contexts:
Context name: MQ QueueConnectionFactoryName: MQSC_QM_QueueConnectionFactory Cached Parameters:
                javax.naming.Context.INITIAL_CONTEXT_FACTORY=com.sun.jndi.fscontext.RefFSContextFactory
                javax.naming.Context.PROVIDER_URL=file:s:\cristi\JNDI
Context name: SERVER QueueConnectionFactoryName:  Cached Parameters:
                BusName=RoutingBus
                ProviderEndpoints=localhost:7286:BootstrapSecureMessaging
                TargetTransportChain=InboundSecureMessaging
Output to routing:
                Point name: Output | Inbound queue: cbetetaaOut | Outbound queue : cbetetaaIn | Context name: SERVER | Automated: true | Enabled :tr
Inputs:
                Point name: MQ connection | Inbound queue: IN | Outbound queue : OUT | Context name: MQ | Automated: true | Enabled :false
                Point name: File connection | Inbound queue: C:\work\workspaces\et_ats\gateway\in | Outbound queue : C:\work\workspaces\et_ats\gate
Nov 05, 2024 10:23:25 AM com.montran.gateway.impl.EtGatewayCoreImpl configureDefaultSSLContext
INFO: configureDefaultSSLContext initializing...
Nov 05, 2024 10:23:25 AM com.montran.gateway.GatewayConnectionHelperWebsphereSIB createConnectionInternal
INFO: Connected to cbetetaaOut
Nov 05, 2024 10:23:25 AM com.montran.gateway.GatewayConnectionHelperWebsphereSIB createConnectionInternal
```

3.2. Interface between Participant Core Banking and Montran Gateway

3.2.1. JMS API Based Interface

This interface provides communication through a pair of input and output queues, that reside on an MQ Server through which payment messages are exchanged between Gateway and the Participant Core Banking

Two queues need to be created in Participant's MQ:

1. One for the messages from the back-office application to the Montran gateway (IN) and
2. One for the messages from the Montran gateway to the back-office application (OUT).

- Queue names and JMS API connectivity details (e.g. JNDI descriptor) are defined when the Gateway is first started. Any JMS compliant server can be used, such as IBM MQ, IMQ, ActiveMQ, etc.

If plain JMS API mode is used then Gateway requires the existence of a **.bindings** file whose location has to be specified in Gateway configuration screen as previously shown.

Bindings file:

The **.bindings** file is created by the Participant (for example when using IBM MQ there is a GUI as well as script option for creating the .bindings file using MQExplorer util).

For other JMS server providers, the **.bindings** file can be created either programmatically using a Java program or via the tools provided by the JMS provider.

The **.bindings** file can be programmatically loaded with the help of **"com.sun.jndi.fscontext.RefFSContextFactory"** initial context. See https://docs.oracle.com/cd/E19340-01/820-6767/aeqbb/index.html

JMS Providers

JMS providers IBM MQ and Glassfish iMQ are supported by default. Other JMS providers also supported if client libraries (each provider has a specific set of needed client libraries) of the provider are copied to the JRE's lib\ext\ directory on the workstation running Gateway.

JMS requirements for exchanged data

When using the JMS interface please note the following requirements:

- Exchanged messages are instances of JMS API TextMessage class.
- The following JMS string properties are mandatory:
  - DESTINATION with value
    - 'RTGS' for high value payments
    - 'FP' for low value payments
  - MSG_TYPE with value
    - 'ISO' for high value payments

        - 'SEPA' for low value payments
    - FILENAME containing the name of the file

Messages received by the Gateway from central systems (ATS/CSD) have other JMS properties but they should be ignored by the Participant's core banking system unless specifically instructed that some property is to be interpreted

3.2.2. File-based Interface

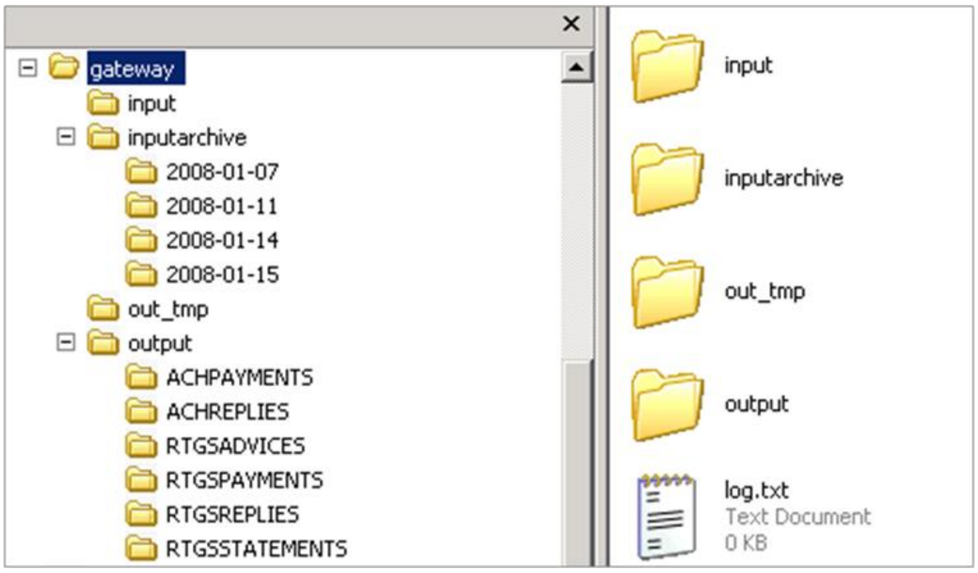
This interface uses two directories IN and OUT (the exact location is defined when the Gateway is first started).

In the IN directory, the back-office application will place files (in ISO 20022 format) and Gateway will scan the directory for new files every few seconds. As soon as a new file is placed in the IN directory, the Montran gateway reads it, validates it and sends it to the ATS system.

In the OUT directory, the Montran Gateway will place the files received from the ATS System.

The participant has the responsibility of transferring the files from the back-office application to the IN directory and taking the received files from the OUT directory.

The folder structure for file connection option is:



```
gateway
    input
    inputarchive
        2008-01-07
        2008-01-11
        2008-01-14
        2008-01-15
    out_tmp
    output
        ACHPAYMENTS
        ACHREPLIES
        RTGSADVICES
        RTGSPAYMENTS
        RTGSREPLIES
        RTGSSTATEMENTS
```

```
input
inputarchive
out_tmp
output
log.txt
Text Document
0 KB
```

Where:

- "input" and "output" are the folders selected in the configuration dialog. "input" folder is used to pick up messages prepared by the Participant Core Banking system. The "output" folder contains several subfolders that will hold the different categories of messages. These folders must be polled by the Participant Core Banking system.
- "inputarchive" and "log.txt" are placed in the same folder as "input". "inputarchive" contains subfolders containing daily archived messages that were received by the CBS and sent to ATS
- "out_tmp" is placed in the same folder as "output". "out_tmp" is for the gateway's internal use

3.3. Message Formats

Messages exchanged between Participant Core Banking system and the ATS System are ISO 20022 IAP1.2 for high value payments (RTGS) and SEPA for low value payments (ACH).

3.4. Message signing technical requirements

There are some technical requirements that must be implemented by Core Banking System in order to support signing/verification of messages.

3.4.1. Overview

All messages exchanged via STP are digitally signed. Montran Gateway can digitally sign the messages itself just before sending to ATS or the Participant Core Banking System can digitally sign the messages before sending them through the Gateway.

3.4.2. E-Token and certificate usage

ATS uses Public Key Infrastructure (PKI) technology to provide non-repudiation. NBG must issue digital signature certificates for the participants to be able to digitally sign messages.

3.4.3. Digital signature format

The exchanged messages are digitally signed using the CMS PKCS#7 standard with SHA256withRSA. This provides authentication, message integrity and non-repudiation of origin.

Please see:

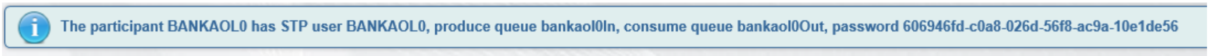
CMS: https://datatracker.ietf.org/doc/html/rfc5652

PKCS#7: https://datatracker.ietf.org/doc/html/rfc2315

For implementation details please visit: https://www.bouncycastle.org/

3.5. Direct API access

The Gateway central system connection infrastructure can be used directly by a participant which wants to integrate more tightly with the ATS. This requires implementing the ATS transport layer protocol using JMS (Java Message Service) API and MQ SIB connectivity model– offered by the central installation. This means that instead of the Java Web Start launcher file, the bank will receive a new session key when using the Gateway option (Routing->Gateway).


The participant BANKAOL0 has STP user BANKAOL0, produce queue bankaol0In, consume queue bankaol0Out, password 606946fd-c0a8-026d-56f8-ac9a-10e1de56

The information included is:

- The jms QueueConnection user
- The jms QueueConnection password
- The jms produce queue name (BANK to ATS)
- The jms consume queue name (ATS to BANK)

The roles of the queues are secured, the produce queue will only accept writes for the specified QueueConnection user, while the consume queue will only accept de-queue operations.

3.5.1. Technical details

In addition to the information in the previous section NBG will provide the **hostname** and the **port** used to initiate connection to ATS.

The following sections list the requirements in connecting to ATS.

Note: The required information can be found on IBM's website: https://www.ibm.com/docs/en/was/9.0.5?topic=ujcwasdmpme-using-jms-resources-thin-client-jms-websphere-application-server

3.5.1.1. WebSphere client libraries

Note: Library version mentioned in this document might change to a newer version if required.

For WebSphere 9 used by ATS, the following client libraries are necessary:

- com.ibm.mq.allclient-9.2.3.0.jar
- com.ibm.ws.messagingClient-9.0.jar
- com.ibm.ws.orb_9.0-9.0.jar
- com.ibm.ws.sib.client.thin.jms_9.0-9.0.jar

3.5.1.2. JMS libraries

- connector-api-1.5.jar
- fscontext-4.6-b01.jar
- jms-1.1.jar
- providerutil-7.5.0.0.jar

3.5.1.3. Security

The API connection to ATS is secured using SSL. A certificate generated by the ATS WebSphere must be used when establishing connections. This certificate will be provided by the NBG on request.

3.5.1.4. Initiating the connection

The following code snippet will initiate a connection to ATS:

```
import com.ibm.websphere.sib.api.jms.*;
...
JmsConnectionFactory jmsCF =
 JmsFactoryFactory.getInstance().createQueueConnectionFactory();
jmsCF.setBusName("RoutingBus ");
jmsCF.setProviderEndpoints("ge_ats_address:7286:BootstrapSecureMessaging ");
jmsCF.setTargetTransportChain(“InboundSecureMessaging”);
```

To obtain a suitable reference to a JMS queue the following code snippet can be used:

```
JmsQueue jmsQ = JmsFactoryFactory.getInstance().createQueue("myQueue");
```

3.5.2. Supported Transport Message Types

The ATS JMS API supports 2 message types:

- javax.jms.TextMessage – used to pass the files as a cleartext string
- javax.jms.BytesMessage – used to pass the files as a byte array (gzipped). When this is used it must be accompanied by the ZIPPED property with value true. This approach saves bandwidth.

3.5.3. JMS Message Header Properties

The messages use the following properties set as a header field

PROPERY NAME	PROPERTY TYPE	DETAILS
DESTINATION	String	The routing destination of the message. When the message is sent by the participant to ATS it should have as value: - FP for ACH messages - RTGS for RTGS messages When the message is sent by the ATS to participant the value is Bank BIC
SOURCE	String	The routing source of the message. When the message is sent from participant to ATS it should have as value the participant BIC When the message is sent by the ATS to particiapnt the value is: - FP for ACH messages - RTGS for RTGS messages
FILENAME	String	The filename of the message.
REF	String	The transport reference of the message. The receiver must check for duplication

RELREF	String	Used only in the confirmation protocol. It takes the value of the original message REF property
DUP	Integer	When a file is being resent (due to lack of timely confirmation), the count of the retransmission.
ZIPPED	Boolean	When the message contents have been compressed with gzip
MSG_TYPE	String	The type of the message: SEPA – ACH business message (.ach.xml files) ISO – RTGS ISO business message (.rtgs.xml files) ACK – delivery confirmation NAK – delivery negative confirmation
CMS	Boolean	When the message content is signed PKCS in a bytes message

3.5.4. Confirmation Protocol

Every message sent by the bank to the ATS or by the ATS to the bank needs to be confirmed. The confirmation is using a special message type (property MSG_TYPE with value ACK) accompanied by the reference of the message being confirmed (property RELREF with value <original message REF>). The body of the message can contain any text.

As an example, the Gateway client uses the following format: **Message exported at <datetime >as User <username> Home Folder <foldername> Java Version <version info> Host info <hostname/hostaddress>**, while the ATS replies with **Message processed**.

For negative confirmation, the NAK value will be set on the MSG_TYPE property and a message body with the reason will be included: **Duplicated delivery**

e.g.
a. Participant (API/Gateway) sends a pacs.008 to ATS
b. ATS accepts the pacs.008 and replies with the above described ACK
c. ATS sends a pacs.002 ACTC back to the Participant

d. The participant accepts the message and sends back to ATS the ACK

etc…