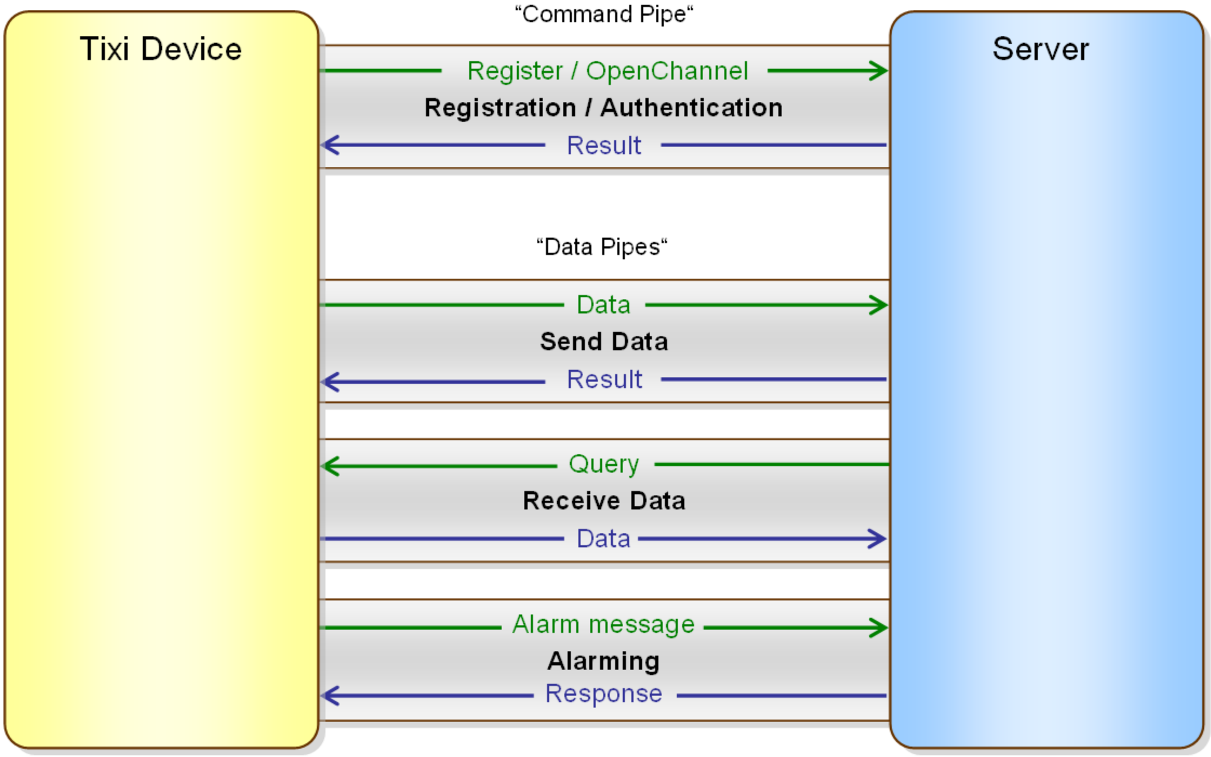
Tixi cloudConn Quickstart - EN



Tixi cloudConnector

Introduction and Quick Start Guide

Release 1.1

Issue date: 14. October 2015

1 Documents

Document	Title
GLT-TiXML--Communication_2014-05-16.pdf	GLT TiXML communication
TiXML-Reference.pdf	TiXML Reference Manual
PLC-TiXML-Manual.pdf	PLC configuration manual

2 Terms Used

Term	Description
TiXML	XML variant for Tixi commands and databases (see also: **TiXML-Reference.pdf**).
XML	**Ex**tensible **M**arkup **L**anguage is a universal language.
XML attribute	A name / value pair within the start tag of an XML element. The name of an attribute must be unique within the XML element. The value is included in "". There are **mandatory attributes**, which need to be used within the start tag of XML elements as well as **optional attributes**, which can be omitted. The value of an omitted attribute will be defined via its default value. Example: `<Element  attributeName=”attributeValue” />`

XML element	A part of XML which is defined through its start tag and end tag.
	`<TagName>`
	`    <embedded tag/>`
	`</TagName>`
	XML elements are identified through its tag names.
	An element can contain other embedded elements. In case there are no embedded elements the short form for start and end tag is used:
	`<ShortTag/>`
	An XML element may contain XML attributes.

TAM TDG	Tixi Alarm Modem Tixi Data Gateway
Tixi IOs	Tixi devices may have - **onboard I/Os** such as digital inputs and outputs, analog inputs and outputs or special IOs etc. or . **pluggable S1-IO-Modules**

Contents

3 Overview: Connecting Tixi to a Cloud Server

The Tixi cloudConnector is a modern protocol to connect Tixi devices with cloud services as easy as possible. It is a client-server protocol and can be implemented by cloud providers with ease.

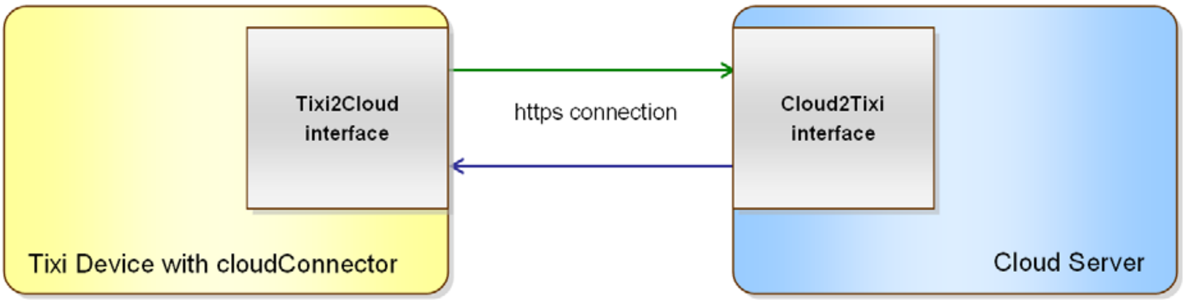


Figure 1: https-Interface between a cloud Server and Tixi

Please note:
A complete documentation of the Tixi cloudConnector protocol for developers is available against NDA upon request to: sales@tixi.com

3.1 Tixi cloudConnector: Communication Pipes

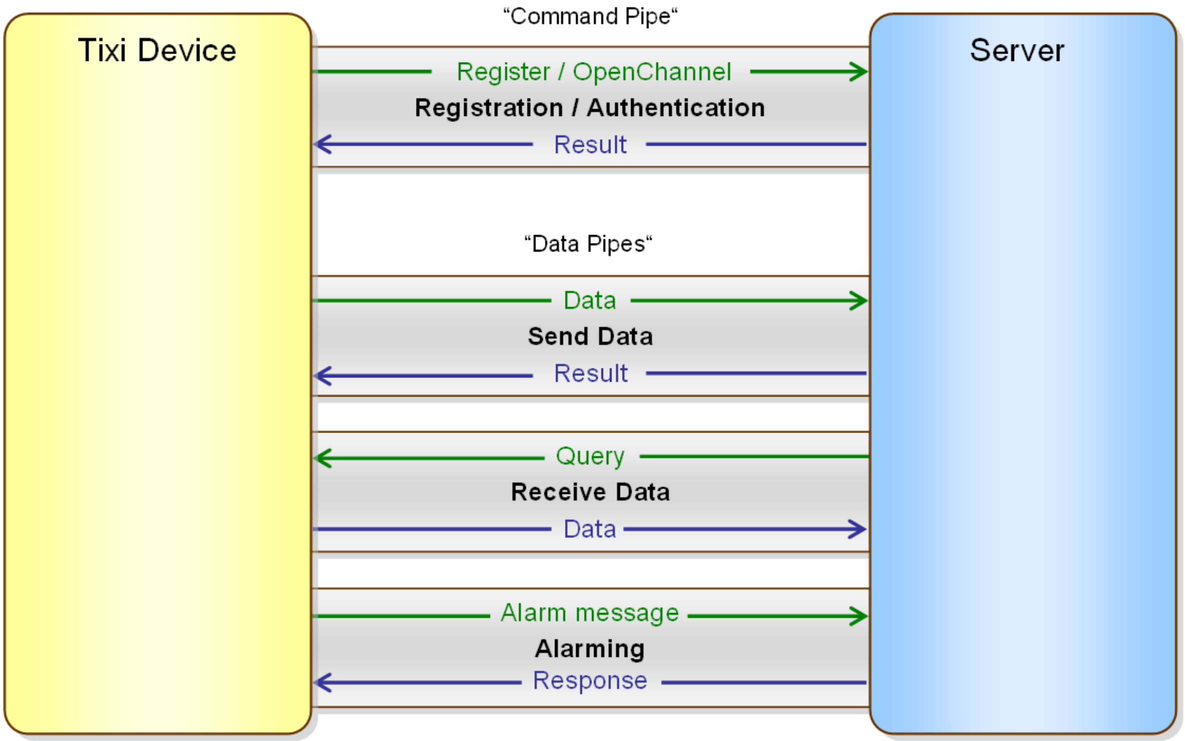


Figure 2: Communication using the Tixi cloudConnector

Tixi establishes a secure **TLS** http connection (also known as SSL) to the cloud server (http-Request) and keeps this connection open. This connection is called "Command Pipe" (CP).

Data sent by the Tixi device is transferred via separate https connections ("Data Pipes"), e.g.:

- Sending responses to server commands
- Sending logfiles to the server
- Sending alarm messages to server

Server and Tixi communicate without delay at any time. "Data Pipes" are closed if the transfer is finished whereas the "Command Pipe" remains open.

Important:
A device needs to be registered at the server before it is allowed to open a "Command Pipe" (authentication request).

3.2 The Tixi Cloud Connector

The Tixi Cloud Connector is a web service built-into Tixi devices which allows you to connect your Tixi device to several cloud service providers like Cumulocity, Telekom "Cloud der Dinge", M2MGo etc.

The protocol of the Cloud Connector is open and can be obtained under NDA. Please contact Tixi.Com GmbH & Co. KG for details.

Requirements
- Tixi device H6xx (Hut line) or W6xx (Wand.Box) with working internet connection
- Existing registration at cloud server (tenant), e.g. Cumulocity
- Working Tixi `LogDefinition` configuration (only if historical data in the cloud are required)
- Working `CloudConn` configuration database

Capabilities of Tixi Cloud Connector
- Send log data to the cloud server
- Send alarm messages to the cloud server
- Send commands and configuration data from the cloud server to the Tixi device (not supported by all servers)

Principle of operation
- Create Tixi device configuration with working Log database (see TiXML reference guide for details)
- Configure the Tixi `CloudConn` database (see chapter 3.3.1)
- Register your account / tenant at cloud service provider (e.g. Cumulocity)
- Register your Tixi device using the Tixi serial number at cloud server (see chapter 3.3.2)

3.3 Connecting your Tixi device with the cloud

3.3.1 CloudConnector configuration database

In order to enable the CloudConnector within the Tixi device a database `ISP/CloudConn` must be configured. The TiXML database is part of the `ISP` database structure.

The configuration can be done with the configuration software TICO or using a config.txt file on a SD card or USB memory stick (depending on the Tixi hardware version).

For Cumulocity servers the CloudConn database is shown below.

```
[<SetConfig _="ISP/CloudConn" >
<CloudConn>
      <CloudBaseUrl     _=" https://tixi-agent.cumulocity.com:443/Tixi" />
      <LogSendTime      _="01.00, 13.00" />
      <RetryTimes       _="2, 5, 10, 30, 60, 300" />
      <CloudConnStart   _="1"/>
      <CloudConnType    _="TixiCloudConnector"/>

      <!-- optional parameters -->
      <username         _="tenant*user" />
      <password         _="password" />
</CloudConn>
</SetConfig>]
```

Mandatory parameters

`CloudBaseUrl`
Base URL to the cloud server

`LogSendTime`
Time(s) at which the Tixi device send all logfiles to the server
Multiple Times may be separated by comma.
Format: `HH.MM` (24 hour format) or M: xx or H: xx

Example 1: send logfiles at 01.00 and 13.00
```
<LogSendTime _="01.00,13.00" />
```

Example 2: send logfiles every hour
```
<LogSendTime _="H: 0-23" />
```

Example 3: send logfiles every 4 hours
```
<LogSendTime _="H: 0,4,8,12,16,20" />
```

Example 4: send logfiles every 10 minutes
```
<LogSendTime _="M: 0,10,20,30,40,50" />
```

Note: It is recommended to use UTC timezone on Tixi device.

`RetryTimes`
Wait time in seconds until the Tixi device retries to open a new "Command Pipe" to the server in case the last attempt failed

`CloudConnStart`
1: Start the CloudConnector at device startup
0: Don't start the CloudConnector at device startup

`CloudConnType`
Type of CloudConnector
"TixiCloudConnector": Tixi protocol as described in this document
"user specific" user specific OEM protocol

Optional parameters
Optional parameters may be omitted completely. If username and password are empty or omitted the Tixi device automatically registers in "Bootstrap Mode" at the server (this is the standard use case).

`username`
Name of user
The username may contain a realm (tenant), e.g. `rwe/name`

`password`
Password (plain text)

Please note:
The Tixi device must be restartet after any change to the CloudConn configuration database.

3.3.2 Register Tixi Device serial number at cloud server

Once the `CloudConn` database has been configured you need to restart the Tixi device.
After the restart the Tixi device is then trying to establish a connection to the cloud server.

If the device was never connected to the cloud server it needs to be registered with the cloud.

The registration process is very simple:

1. Find out the serial number of your Tixi device
   The 8-digit serial number can be found on a sticker. It usually starts with 04
2. Power on your Tixi device and make sure that it has a working internet connection (LAN or GPRS)
3. Log into your cloud server and go to the device registration page.
4. Enter the serial number of the Tixi device (8 digits)
5. After about 1 .. 2 minutes the Tixi device should be automatically registered with the cloud server.

6. Depending on the cloud service (e.g. Cumulocity) you may need to manually "Accept" the serial number on the web pages once the device has established the connection.
   Please consult your cloud service provider for details.

During the registration process the LED labeled "Signal" is blinking fast in red color.

As soon as the registration process with the cloud has been finished the "Signal" LED is blinking green (slowly) and after a few seconds it is switched on in green color permanently.

Your device is now securely connected to the cloud !

3.3.3 Re-Register Tixi Device serial number at different cloud server / tenant

In case you want to connect your Tixi cloud device to a different cloud service or a different tenant at your cloud provider you need to de-register your Tixi device with your current cloud provider:

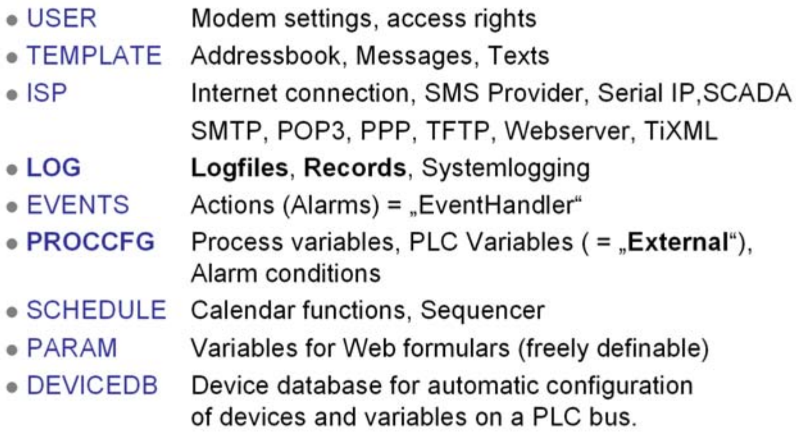
1. Delete the device from your device list on the web page of your current cloud provider
2. Send the following command to your Tixi device (e.g. with TICO software):
   ```
    [<Reset _="Secretfile" ver="y" />]
   ```
3. Change your `CloudConn` database if neccessary (only if you want to change to a different cloud service provider)
4. Reboot your Tixi device

Now you may register your device again with another cloud service as described in chapter 3.3.2.

4 Introduction to Tixi Gateways

Tixi gateways currently support up to 6 different local bus interfaces, e.g. 2x RS232, 2x RS485, M-Bus and 1-Wire and one local Ethernet port, e.g. for a PLC.
Depending on the local bus protocol each bus interface can communicate with hundreds of devices on that bus (PLCs, energy meter, sensors etc.).

The Tixi system uses a set of XML databases for configuration of the system:



● USER	Modem settings, access rights
● TEMPLATE	Addressbook, Messages, Texts
● ISP	Internet connection, SMS Provider, Serial IP,SCADA SMTP, POP3, PPP, TFTP, Webserver, TiXML
● **LOG**	**Logfiles**, **Records**, Systemlogging
● EVENTS	Actions (Alarms) = „EventHandler“
● **PROCCFG**	Process variables, PLC Variables (= „**External**“), Alarm conditions
● SCHEDULE	Calendar functions, Sequencer
● PARAM	Variables for Web formulars (freely definable)
● DEVICEDB	Device database for automatic configuration of devices and variables on a PLC bus.

In addition to TiXML commands for **controlling** Tixi
such as Set, Get, ReadLog etc.,
TiXML commands are used for **configuring** Tixi:

"**SetConfig**" to store and
"**GetConfig**" to retrieve these TiXML configuration databases.

A Tixi device reads ("polls") data from the locally connected devices using a specified poll rate and stores it in its internal "Process data memory". This data is called "Process Data". The "Process Data" contains the results of the latest reading from such external devices, according to the specified poll rate.
The configuration of such external protocols is done in the database "`External`".

A second process called "Logging" stores selected Process Data into log files. The Logging is triggered by scheduler or system events. Both processes (collecting "Process Data" and "Logging") are completely independent. The Logging configuration is done via database "`LOG/LogDefinition`"

The configuration of the Tixi cloudConnector is stored within a TiXML database called "`CloudConn`".

```
[<GetConfig _="ISP/CloudConn" >

<CloudConn>
      <CloudBaseUrl     _="https://mycloud.mydomain.com/Tixi" />
      <LogSendTime      _="01.00, 13.00" />
      <RetryTimes       _="2, 5, 10, 30, 60, 300" />
      <CloudConnStart   _="1"/>
      <CloudConnType    _="TixiCloudConnector"/>

      <!-- optional parameters -->
      <username         _="tenant*user" />
      <password         _="password" />
</CloudConn>

</GetConfig>]
```

See chapter 3.3.1 for a detailed description of the `CloudConn` database.

4.1 Important Databases

4.2 <External> Database

The configuration of the bus interfaces is stored in the XML database "`PROCCFG/External`".
Path: `/SYSTEM/CONFIG/PROCCFG/External`
Live example:
`http://193.101.167.68:8080/SYSTEM/CONFIG/PROCCFG/External`

Database structure

```
<External>
    <bus definition for bus 1>

        <Device 1>
            <Variable definitions, one variable at one line/>
        </Device 1>

        <Device 2>
            <Variable definitions, one variable at one line/>
        </Device 2>

    </bus end of definition for bus 1>

    <bus definition for bus 2>
        <Device 1>
            <Variable definitions, one variable at one line/>
        </Device 1>
        <Device 2>
            <Variable definitions, one variable at one line/>
        </Device 2>
    </bus end of definition for bus 2>

    … more bus definitions

</External>
```

There are many options available for the `External` database, depending on the bus protocol and variable types.
A comprehensive description can be found in the document "`PLC-TiXML-Manual.pdf`".

4.3 Current Values of Local Data Points DP

The term “data point” (DP) is used for any local user data point, such as those variables declared in the “`External`” data base for the variables of each device on a bus interface.

These device variables from the local bus interfaces are being mapped to the TiXML variables in the “`External`” data base definition according to the poll rate of the respective devices.

The current value of local DPs can be retrieved as follows:

all DPs on all devices of all busses
`http://193.101.167.68:8080/SYSTEM/PROPERTIES/Process/`

all DPs on all devices of ONE bus
`http://IP/SYSTEM/PROPERTIES/Process/bus/`

all DPs on ONE device of ONE bus
`http://IP/SYSTEM/PROPERTIES/Process/bus/device`

ONE DP on ONE device of ONE bus
`http://IP/SYSTEM/PROPERTIES/Process/bus/device/DP-name`

4.4 Process Variables

Process Variables are freely definable variables in TiXML (global variables).
The value of a process variable can be changed by I/O ports, PLC variables, logical or mathematical operations or via trigger conditions.
The definition is made in database "`PROCCFG/ProcessVars`".

The current value of all Process Variables is being updated constantly.
They can be retrieved as follows:
`http://193.101.167.68:8080/SYSTEM/PROPERTIES/Process/PV/`

4.5 <LOG> database

The configuration of the logging system is made in the XML database "`LOG/LogDefinition`".
You can store Process Data and Process Variables ("/Process/" tree) in a log file.

Path: `/SYSTEM/CONFIG/LOG/LogDefinition`
Live example:
`http://193.101.167.68:8080/SYSTEM/CONFIG/LOG/LogDefinition`

Database structure

```
<LogDefinition>

    <Logfiles>
        <!-- system log files (names may not be used for user log files) -->
        <JobReport   size="FSize" />
        <Event       size="FSize" />
        <Login       size="FSize" />
        <SupportLog  size="FSize" />

        <!-- user log files -->
        <FileName1   size="FSize" [record="Record1"] contenttype="Ctype"] />
        <FileName2   size="FSize" [record="Record2"] contenttype="Ctype"] />

    </Logfiles>

    <Records>

      <Record1>
        <VarName1 _="VType" Name="UserVarName1" path="Source" size="Size" />
        <VarName2 _="VType" Name="UserVarName2" path="Source" size="Size" />
        <VarName3 _="VType" Name="UserVarName3" path="Source" size="Size" />
      </Record1>

      <Record2>
        <VarName1 _="VType" Name="UserVarName1" path="Source" size="Size" />
        <VarName2 _="VType" Name="UserVarName2" path="Source" size="Size" />
        <VarName3 _="VType" Name="UserVarName3" path="Source" size="Size" />
      </Record2>

    </Records>
```

FSize	Size of file in bytes
Record1	Name of record definition for this log file: A record definition is a list of variables to be stored in a log file
Ctype	Content type: `binary` (default) or `xml`
Source	Path to Process Data tree, e.g. `/Process/Bus1/Device_1/Variable_1`
VType	Variable type
Size	Size of variable in bytes (depends on data type)

For system log files only the option “size” can be defined. Usually only user log files will be transferred to the server. IOW: All log files which have a tag `record=` are definable user log files.

A description of the `LogDefinition` database can be found in the document "`TiXML-Reference.pdf`"

4.6 TiXML Example

We have defined one bus named "M-Bus" on interface "COM3",
protocol = "Meterbus",
baudrate = 2400, handshake = none,
Tixi device is the bus master.

This bus has an electricity meter named "DHZ-1" with a poll rate of 1 second.
The primary address of this meter is 62.
It has 5 variables: Energie, Spannung, Strom, Leistung, Seriennummer

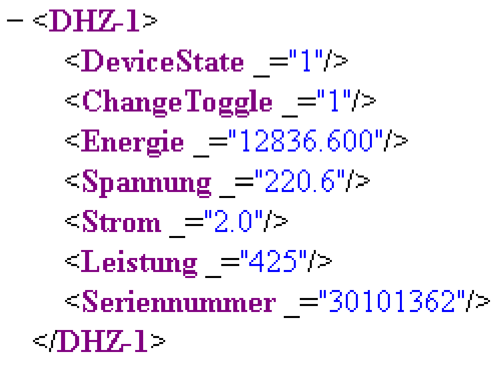
```
<External>

      <Bus _="COM3" family="M-Bus" Product="M-Bus Standard"
            protocol="Meterbus" Mem="123000"
            baud="2400" handshake="none"
            type="Master"
            AddProperties="Name,TimeStamp">

            <Device _="0" Name="DHZ-1" NameUser="DHZ-1" Pollrate="1s" PrimaryAddr="62">

                  <Energie _="value" Name="Energie" simpleType="meterbus" ind="1"
                        size="13" exp="-3" acc="R"/>

                  <Spannung _="value" Name="Spannung" simpleType="meterbus" ind="3"
                        size="13" exp="-3" acc="R"   format="F.1;"/>

                  <Strom _="value" Name="Strom" simpleType="meterbus" ind="4"
                        size="13" exp="-3" acc="R" format="F.1;"/>

                  <Leistung _="value" Name="Leistung" simpleType="meterbus" ind="5"
                        size="13" acc="R" def="-1"format="{=-1};{};%%"/>

                  <Seriennummer _="value" Name="Seriennummer" simpleType="meterbus" ind="6"
                        size="13" acc="R"/>
            </Device>

      </Bus>
</External>
```

The Process Data of the defined busses and devices can now be retrieved using the following paths
`/Process.`
Live example:
http://193.101.167.68:8080/SYSTEM/Properties/Process/

Process Data of bus “M-Bus”
`/Process/M-Bus`
Live example:
http://193.101.167.68:8080/SYSTEM/Properties/Process/M-Bus/

Process Data of the first device ("DHZ-1") on M-Bus
`/Process/M-Bus/DHZ-1`
Live example:
http://193.101.167.68:8080/SYSTEM/Properties/Process/M-Bus/DHZ-1/

Response from live example:



```
- <DHZ-1>
    <DeviceState _="1"/>
    <ChangeToggle _="1"/>
    <Energie _="12836.600"/>
    <Spannung _="220.6"/>
    <Strom _="2.0"/>
    <Leistung _="425"/>
    <Seriennummer _="30101362"/>
  </DHZ-1>
```

Now we want to add some data points of this device into our log file.
We create the following LogDefinition:

```
 <LogDefinition>

    <Logfiles>

        <!-- system log files (names may not be used for user log files) -->
        <JobReport   size="20000" />
        <Event       size="20000" />
        <Login       size="20000" />
        <SupportLog  size="20000" />

        <!-- user log files -->
        <Logfile1 size="50000" record="Energy_1" contenttype="binary" />

    </Logfiles>


    <Records>

      <Energy_1>
        <Energie_DHZ-1 _="meterbus" Name="Energy Server"
             path="/Process/M-Bus/DHZ-1/Energie" size="13" />

        <Spannung_DHZ-1 _="meterbus" Name="Spannung Server"
             path="/Process/M-Bus/DHZ-1/Spannung" size="13" />

        <Leistung_DHZ-1 _="meterbus" Name="Leistung Server"
             path="/Process/M-Bus/DHZ-1/Leistung" size="13" />
      </Energy_1>

    </Records>

</LogDefinition>
```

The log file "`Logfile1`" now contains 3 variables:

`Energie_DHZ-1, Spannung_DHZ-1, Leistung_DHZ-1`

Each record entry of the log file has its own ID number and time stamp.

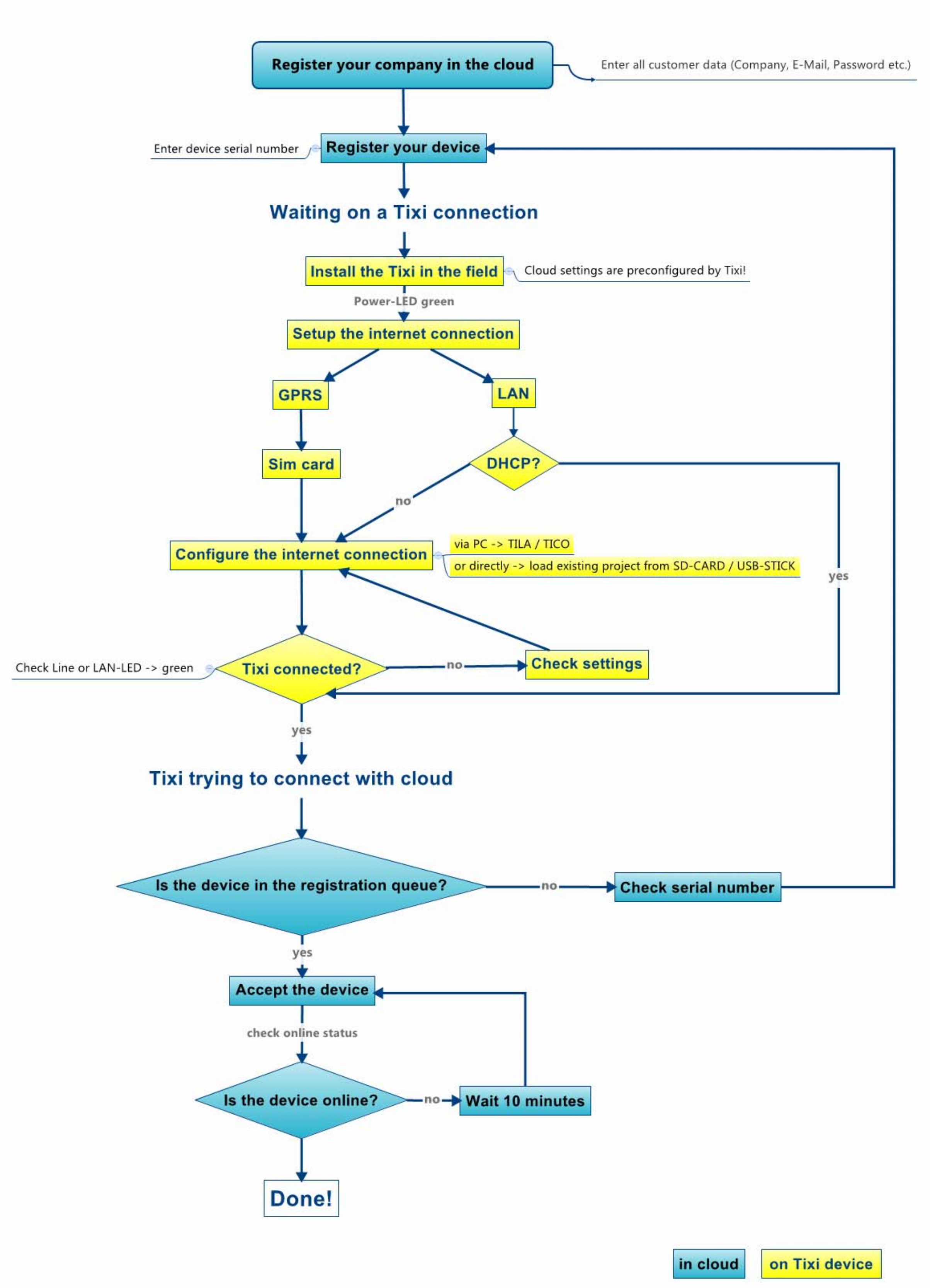
Log file entries will be created either

via scheduler (e.g. one entry every 15 minutes)
or with a trigger command.

You may define up to 100 user log files with different file sizes, log record structures and trigger events.

5 Appendix A

5.1 Scheme: Connect Tixi devices with the Cumulocity cloud


Register your company in the cloud
Enter all customer data (Company, E-Mail, Password etc.)
Enter device serial number
Register your device
Waiting on a Tixi connection
Install the Tixi in the field
Cloud settings are preconfigured by Tixi!
Power-LED green
Setup the internet connection
GPRS
LAN
Sim card
DHCP?
no
yes
Configure the internet connection
via PC -> TILA / TICO
or directly -> load existing project from SD-CARD / USB-STICK
Check Line or LAN-LED -> green
Tixi connected?
no
Check settings
yes
Tixi trying to connect with cloud
Is the device in the registration queue?
no
Check serial number
yes
Accept the device
check online status
Is the device online?
no
Wait 10 minutes
Done!
in cloud
on Tixi device

5.2 List of Figures

Version history

Version	Date	Author	Comment
1.0	16.07.2015	SH	Release
1.1	14.10.2015	SH	Some chapters re-arranged