





Table of Contents

1. Universal Data Driver guide	3
2. How to configure UDD text channel	14
3. UDD4Demo.xml	29



1. Universal Data Driver guide

Universal Data Driver Idea

Universal data driver (UDD for short) is a configurable text data capture and transaction management interface for Mirasys VMS.

It has several communication interfaces to interact with text data sources, namely TCP/IP and HTTP client and server, and UDP/IP and serial (RS-232) server protocols. There is one special case for RTSP transaction messages. It is used for UDP technologies' IP camera VCA metadata event capturing.

Transaction messages can be either text strings (ASCII messages) or XML formatted messages.

Text format validation and parsing are done using regular expressions (Boost RegEx). XML messages are validated using XSD and parsing using XPath notation. For binary messages and transaction sources requiring proprietary parsing and special communication protocol (for example SIA or ContactID protocols), there is a possibility to use custom validation (pre-processing) in custom DLL (dynamic link library).

All channels support UTF-8 and UTF-16 text as well as binary data formats (custom validation).

As a result transaction events can trigger alarms. By default, all transaction messages are stored in a text data channel.

Text data content can later be searched using the Spotter text search plugin and be used to synchronize video and audio playback.

COMMUNICATION LAYER CLASS DIAGRAM



Mirasys Universal Data Driver



DATA VALIDATION AND MAPPING INTERFACE

STRUCTURE OF CONFIGURATION FILE FOR SIMPLE TEXT ALARM DATA

<?xml version="1.0" encoding="UTF-8"?> <root>

This example describes validation and message mapping configuration for text data alarms that a Piccolo alarm system can send to the driver.

CHANNEL CONFIG

Channel Config section defines the **linefeed** value of the incoming messages, i.e. the message separator.

Mirasys LtdC1CD, Vaisalantie 2-8, 02130Espoo, FinlandTel +358 (0)9 2533 3300-info@mirasys.com-www.mirasys.com



Typically Windows applications terminate messages to carriage return and linefeed (CR+LF = 0x0D0A). Unix systems (like IP cameras) typically use only linefeed (LF = 0x0A).

In some cases messages are not separated, i.e. source system is sending data in bursts.

In that case, the linefeed value should be left empty ("") and use text type parsing to match the incoming character stream.

UDD channel reader can ignore selected characters or byte values to clean the data and help to parse.

Tag **ignored** lists all the characters that should be filtered out before passing them to the UDD parsing phase.

The most common ignored value is a NULL byte, i.e. 0x00.

When the text channel is active and the text data window is open in Spotter, all incoming messages are echoed in it.

If the user would like to control the view, it is possible to clear the window when needed.

This can be done by sending the **clearscreen** string to the text channel.

<channelConfig> <linefeed value="0x0d0a"/> <ignored value="0x00"/> <clearscreen value="-----"/> </channelConfig>

Validation

Different validation methods are described under a validation element. The given attribute for regex contains a regular expression string to be used to validate text data ("*" at the end means 0 or many of the before defined character set).

<validation> <regex value=".*"/> </validation>

For XML formatted messages XSD (XML schema) validation type is used:

<validation> <xsd value="MySchema.xsd"/> </validation>



If the .xsd file is defined, this file will be used for incoming XML message validation. But the "value" attribute can be empty - in this case, the common validation will be used: only checking that the incoming message has the correct XML format.

Log messages in the DVRLog.txt

Detailed logging mode can be enabled by inserting the following option in the XML configuration file.

Logging level parameters are:

- 0: no writing to dvrlog.txt
- 1: only error messages, error and info messages.

To enable additional message writing and input data packet storing, you should add

<additionalDebug> tag to logging element with the attribute value="yes". Data packets read into the text channel will be written into a DVR folder file named "Debug_UDD_Packets.bin".

Additional text channel info will be written into the file "Debug_UDD_Log.txt" in the same folder.

<logging> <level value="2"/> <additionalDebug value="yes"/> </logging>

MESSAGE CONTENT PARSING – UDDXMLMAPPER ELEMENT

Transaction event message content parsing is defined in **uddXmlMapper** element. Incoming messages can be ASCII strings or well-formed XML. Binary messages will need pre-processing before UDD can handle them. Typically pre-processing from binary format to internal XML format is done in a separate proxy service (for example PaxtonProxy or BoschIVAproxy) or in UDD custom validator (for example GalaxyDataValidator.DLL or TexecomDataValidator.DLL).



<uddXmlMapper version="2">

Message definitions – messageType element

Inside uddXmlMapper element first sub-element is **messageType**. The message type element defines the incoming message content format (ASCII **text** or **XML**).

<messageType value="text" parsing="regex">

Inside message type elements are messages with their content references. ASCII text parsing tagging is defined using regular expressions and XML parsing using XPath notation to reference tags ("+" at the end of a group means 1 or many of the before defined character set).

```
<messageType value="text" parsing="regex">
<message number="1" value="alarm">
<param number="1" value="([a-zA-Z]+) (.+)" group="1"/>
<param number="2" value="([a-zA-Z]+) (.+)" group="2"/>
</message>
</messageType>
```

In the above example, ASCII text has three interesting parts; a string of letters, a string of any characters and a string of numbers.

Strings are separated by spaces.

If the incoming message would be (followed by the defined line feed)

then

group="1" match "start" group="2" match "low" group="3" match "4" rest of the message would be ignored from a matching point of view. If the incoming XML message having the same kind of information would be (followed by the defined line feed)

```
<?ml version="1.0" encoding="UTE-8"?>
<alarm>
<trule>start</rule>
<type level="4">low</type>
<info>this is normal</info>
</alarm>
```

then

"alarm/rule" match "start" "alarm/type" match "low" "alarm/type/@level" match "4" Note that XML tag attributes are denoted using @ sign in front of the attribute name.

 Mirasys Ltd
 C1CD, Vaisalantie 2-8, 02130
 Espoo, Finland

 Tel +358 (0)9 2533 3300
 info@mirasys.com
 www.mirasys.com



Message content constants – Constants element

These constant arrays contain values that can be used in the message handling rules section.

<constants></constants>
<array name="Rules" params="AT"></array>
<value>start</value>
<value>stop</value>
<array name="AlarmType" params="AT"></array>
<value>ACCESS CONTROL ALARM 1</value>
<value>ACCESS CONTROL ALARM 2</value>
<value>ACCESS CONTROL ALARM 3</value>

In this example, incoming messages contain "Rules" strings that have values "start" and "stop".

Params "AT" and "TL" mean that these values will be used at least in two different kinds of rule definitions in the UDD rules element.

Action rules - Rules element

Action rules, i.e. how to handle incoming transaction messages, are defined in Rule's element.

There can be more than one type of message and its rules, but typically UDD is configured to handle messages only in one format.

However, one message format usually has several rules and events defined.

```
<rules>
<message number="1" value="alarm" alwaysShowText="yes">
<param number="1" reference="1" value="$Rules" operator="eq" id="AT">
<and number="1" reference="2" value="$AlarmType" operator="eq"/>
<action number="1" type="event" value="$Rules $AlarmType alarm" />
</param>
</message>
</rules>
```

In this example <message number="1" value="alarm" refers to the messageType element and its message definition.

```
Mirasys LtdC1CD, Vaisalantie 2-8, 02130Espoo, FinlandTel +358 (0)9 2533 3300-info@mirasys.com-www.mirasys.com
```



Note, that in text format messages the message tag attribute in messageType has to have a name (value="alarm").

In XML format messages this value is by default name of the root tag, in this example alarm (value="**alarm**/rule").

Message tag attribute alwaysShowText="yes" means that all incoming messages will be shown in the text channel device window.

Message tag attribute handleParameters=" all" means that for the incoming message all rules are run through. By default, UDD stops to the first matching rule. If there were more than one message format handled in the rules element, then message elements will have an ascending number attribute number="1", number="2", etc.

Rules for the message are defined in the param element. Param elements have an ascending number attribute.

Param element reference attribute is a sequence number of message tags (see messageType/message element above, in the example group="1" match "start" or "alarm/rule" match "start"). So in the example above reference="1" would contain the value "start".

Param tag attribute can be either constant value="start" or a reference to constant array (in constants element) value="\$Rules".

Using constant arrays UDD automatically expands rules using the values of constant arrays during runtime.

So, the operation would be the same (in the above example) either using arrays (Rules array containing strings "start" and "stop") or manually configuring two rules params

<param number="1" reference "1" value="start" operator="eq"> <param number="2" reference "1" value="start" operator="eq">

Comparison operators that can be used as Boolean operators are:

eq	= [defmlt]
qt	>
ge	
16	< c
le	-
neq	!= not equal
contains	finds configuration parameter value as substring in the incoming message
	parameter value
contained	finds incoming message parameter value as substring in the configuration
	parameter value, this operator is supported in the 2.13.6.0 version of the UND
	or higher

Each parameter can have the "type" attribute which is used for correct text values conversion for comparing. Currently, the following types are supported:

- type="integer"
- type="float"

Mirasys Ltd - C1CD, Vaisalantie 2-8, 02130 - Espoo, Finland

Tel +358 (0)9 2533 3300 - info@mirasys.com - www.mirasys.com



• type="string"

If the "type" attribute is not specified, the "string" type is used by default. The rule per message can contain one or more parameters, and these parameters can refer to one or more parameters defined in the messageType element as described above.

This example configuration XML for a simple text alarm data has only one parameter per message, so message parameters in message rules refer to parameter 1.

```
<rules>
<message number="1" value="Alarm cam001">
<param number="1" reference="1" type="string" value="Alarm
cam001">
```

Zero or more actions can be associated with message parameters. Currently, defined actions are "event"

to trigger events, "data" to send data, and "metadata" to send metadata to DVR.

<action number="1" type="event" value="Alarm cam001" />
<action number="2" type="data" value="Alarm cam001 data" />
<action number="3" type="metadata" value="" />

For metadata one additional attribute can be used - "metadatatype". If it is specified it allows to configure metadata type:

- "xml" XML metadata format
- "udpvca" special UDP VCA metadata format
- "textevent" text event metadata format
- "other" other metadata formats

If the "metadatatype" attribute is not specified the default text channel format is used.

The parameters can also have an associated search tag. If a search tag type attribute has been omitted, it is set by default to be SEARCH_TAG_TYPE_TEXT.



```
<searchtag value="Alarm cam001" />
        </param>
      </nessage>
      <message number="2" value="Reset 001">
        cparam number="1" reference="1" type="string" value="Reset"
001">
          <action number="1" type="event" value="Reset 001" />
          <searchtag value="Reset 001" />
        </param>
      </message>
      <message number="3" value="Alarm cam002">
        <param number="1" reference="1" type="string" value="Alarm"</pre>
cam002">
          <action number="1" type="event" value="Alarm cam002" />
          <searchtag value="Alarm cam002" />
        </param>
      </nessage>
      <message number="4" value="Reset 002">
        <paran number="1" reference="1" type="string" value="Reset"</pre>
002">
          <action number="1" type="event" value="Reset 002" />
          <searchtag value="Reset 002" />
        </param>
      </nessage>
    </rules>
  </uddXmlMapper>
</root>
```

So what this example configuration will do is to tell the driver that:

- we are expecting incoming data to be textual
- text data must match with the defined regex pattern
- there are four possible message types

For example, if the incoming data is

Alarm cam001

it is validated against regex and matched against defined message types. The match is found with the defined message number 1, so the driver will send an event and "Alarm cam001" data to DVR. If the incoming data would be

Alarm cam100

it would validate against regex but no matching rule is found, so it would be ignored.

And if the incoming data would be

Alarn #8s%#!



it would not validate against a defined regex pattern and would generate an error in the DVR log file.

WHEN STARTING THE INTEGRATION TEST

It is a best practice that before taking the transaction message validation and parsing configuration into use, first test the communication between the two systems.

This way all captured messages will be written directly into the text channel. Select validation mode "None" in the channel configuration dialogue, start transaction counterpart and monitor text channel device window in Spotter user interface.

ADDITIONAL PARAMETERS FOR HTTP SERVER

Additional parameters can be configured for the HTTP server channel in the XML configuration file.

These parameters should be placed to the special section

<HTTPServer></HTTPServer>. Currently, the following parameters are supported:

- "processingType" attribute the special attribute to determine what part of incoming HTTP request will be processed:
 - All whole message
 - Headers only HTTP headers from the incoming message
 - Uri only HTTP URI string from the incoming message
 - Content only HTTP content from the incoming message
- <Response> element the HTTP response message content.
 - If the attribute "send" value is no, the driver will send an HTTP response message without content
 - If the attribute "send" value is yes, the driver will send an HTTP response message with specified content data

Example of additional parameters configuration for HTTP server channel:



(Additional)
diffester processingType="All">
<pre>«Response send="tzue">Response staring to send(/Response)</pre>

- type="integer"
- type="float"
- type="string"

If the "type" attribute is not specified, the "string" type is used as default.

param number="1" reference="1" value="\$Mules" operator="eg" id="AT"> -(and number="1" reference="2" value="\$AlarmType" operator="eg"/> -(action number="1" type="event" value="\$Mules \$AlarmType alarm" /> </param>

It is possible to combine more than one content reference using Boolean operator AND. In the example

message content reference="1" ("alarm/rule") and reference="2" ("alarm/type") must both be true ("and" operator) in order to trigger the action event value="\$Rules \$AlarmType alarm".

If the message content would have been for example "alarm/rule" matching "start" and "alarm/type" matching "low", then the trigger "start low alarm" would have been triggered.



2. How to configure UDD text channel

Create UDD configuration file(s); for text validation UDD configuration XML and for XSD validation both UDD configuration file and XSD validation schema. **Hint**: good online XML tools can be found from <u>http://www.freeformatter.com/xml-validator-xsd.htmlhttp://www.freeformatter.com/xsd-generator.html</u> Copy configuration file(s) into DVR folder (for example C:\Program Files (x86)\DVMS\DVR.

Adding text channel to the Mirasys VMS

- 1. Open System Manager
- 2. Go to the VMS servers tab
- 3. Open Text channels



- 4. Select first free text channel
- 5. Click Add channel



•	Text	Channel Settings		×
	•	DahuaAPIProxy	Driver: UniversalDataProxyModel	^
	•	ASSA ABLOY ARX	Driver: UniversalDataProxyModel	
	۰	UDD Test Channel	Driver: UniversalDataTcpModel	
	•	Text Channel 4		
	Þ	Text Channel 5		
	•	Text Channel 32		~
			Add char	nnels

- 6. Set Model to UniversalDataTcpModel
- 7. Enter needed **TCP Port number**(default **40000**)
- 8. Set Validation to Text
- 9. Click **OK**



💽 Edit Text Channel	×
Model: First channel: Number of text channels:	UniversalDataTcpModel 3 1
- Properties	
TCP Port number	40000
Validation	Text V 8
Configuration file	UDD4Demo.xml
Custom validator	
Send the "End" event after N	
Forward incoming message to	
	×

10. Enter the name of the text channel and press **OK**.



•	Text	Channel Settings	5				>
	Þ	DahuaAPIProx	y	Driver: Un	iversalDataProxyModel		^
	Þ	ASSA ABLOY A	RX	Driver: Un	iversalDataProxyModel		
	•	UDD Test Chan	nel	Driver: U	niversalDataTcpModel		
	•	UDD Test		Driver: U	niversalDataTcpModel		
		Name:	UDD Test				
		Description:	General description		Administrative Description		
				~		^	
				\sim		~	
Γ		Driver:	<u>UniversalDataTcpi/c</u>				
Γ		in use					

Editing UDD4Demo.xml

- 1. Open text editor with admin rights
- 2. Browse to C:\Program Files\DVMS\DVR
- 3. Select UDD4Demo.xml file
- 4. Edit

<array name="Rules" params="AT"> <value><mark>START</mark></value> <value><mark>STOP</mark></value>

5. Edit

<array name="AlarmType" params="AT"> <value>ACCESS CONTROL 1 EVENT <value>ACCESS CONTROL 2 EVENT <value>ACCESS CONTROL 3 EVENT </value>

- 6. Save changes and close text editor
- Open created text channel and rename text channel and click **OK(**Mirasys VMS reloads changes from UDD4Demo.xml)

In this example, we have edited the UDD4Demo.xml file so that when the below text appears, that will be used as a trigger of the alarm. All other texts are ignored.



START ACCESS CONTROL 1 EVENT START ACCESS CONTROL 2 EVENT START ACCESS CONTROL 3 EVENT

Creating an alarm from the validated text

- 1. Go to the VMS Servers\Alarms
- 2. Click New Alarm
- 3. Enter the name of the alarm(in this example we are using **START ACCESS CONTROL 1 EVENT, START ACCESS CONTROL 2 EVENT and START ACCESS CONTROL 3 EVENT**)
- 4. Set Priority and View alarm in profiles



o Alarm Co	onfiguration						
General	Trigger Actions Calendar						
	START ACCESS CONTROL 1 EVENT	Vi	View alarm in profiles:				
	Description Administrative Description		Visible	Profiles			
	· · · · · · · · · · · · · · · · · · ·		<u></u>	Demo			
				Mirasys A∨M			
	<u> </u>	L		Koulutus			
	Priority						
	High						
	Normal						
	Low						
	C ^{Options}						
	The alarm is active until it is acknowledged						
	Alarm highlight color						
	Use default color						
	O Use custom color -						
		1					
		1					
		- 11					
		1					
		- 11					
		- 11					
		- 11					
		- 11					
		- 11					
		- 11					
		- 11					
		- 11					
		- 11					

- 1. Open Trigger tab
- 2. Select **Text data** from the dropdown menu





- 6. Select correct text channel
- 7. Select alarm trigger from the upper box(this example **START ACCESS CONTROL 1 EVENT**)
- 8. Enable **Define ending input,** if needed(If Define ending input is set, the alarm will be active until selected input text is received to the text channel)
- 9. Select correct ending input(this example STOP ACCESS CONTROL 1 EVENT)



of Alarm	Configuration					
General	Trigger Action	s Calendar				
Туре:	Text data	~	6		•	۹.
	DahuaAPIProxy	ASSAABLOY ARX	UDD4 DEMO	Text Channel 4	7	Connection lost START ACCESS CONTROL 1 EVENT alarm START ACCESS CONTROL 2 EVENT alarm START ACCESS CONTROL 3 EVENT alarm STOP ACCESS CONTROL 1 EVENT alarm STOP ACCESS CONTROL 2 EVENT alarm STOP ACCESS CONTROL 3 EVENT alarm
	Text Channel 5	Text Channel 6	Text Channel 7	Text Channel 8	I	
	Text Channel 9	Text Channel 10	Text Channel 11	Text Channel 12	I	
	Text Channel 13	Text Channel 14	Text Channel 15	Text Channel 16	8	Define ending input
	Text Channel 17	Text Channel 18	Text Channel 19	Text Channel 20	9	START ACCESS CONTROL 2 EVENT alarm START ACCESS CONTROL 3 EVENT alarm STOP ACCESS CONTROL 1 EVENT alarm STOP ACCESS CONTROL 2 EVENT alarm STOP ACCESS CONTROL 3 EVENT alarm

1. Open **Actions** tab

2. Select actions of the alarm from the dropdown menu and add them to the **Visible** box by clicking **Add**





- 1. Open **Calendar**
- 2. Define, alarm active hours
- 3. Click **OK**



of Alarm Configuration									
General Trigger Actions Calendar									
Regular Schedule Exception days									
Off		Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday	
	0 ap.	On	On	On	On	On	On	On	
	1 ap.								
	2 ap.			-		2 2			
	3 ap. 4 ap.	-				2 2		-	
	5 ap.								
	6 ap.								
	7 ap.								
	8 ap.	-				-			3
	эар. 10 ар								
	11 ap.					-			
	12 ip.					8			
	13 ip.								
	14 ip.	-							2
	15 lp. 16 in		1						
	17 ip.		i i i i i i i i i i i i i i i i i i i						
	18 ip.					1			
	19 ip.		Î.						
	20 ip.								
	21 ip.					8 <u>.</u>			
	22 ip. 23 in.								
	20 (0.								
									/ ×

We have now created a separate alarm for each event. Below is an overall view of each alarm.

START ACCESS CONTROL 1 EVENT





START ACCESS CONTROL 2 EVENT



START ACCESS CONTROL 3 EVENT

VI START ACCES	SS CONTROL 3 EVENT	Normal	Text data in UDD4 DEMO	Record video from BOSCH FLEXIDOME
Name:	START ACCESS CONTROL 3 EV	ENT		
Description:				
Priority:	Normal			
Requires Acknowledgment:	No			
Viewable in Profiles:	<u>Demo</u>			
Trigger:	Text data in UDD4 DEMO Activate at: START ACCESS CO Deactivate at: STOP ACCESS CO	NTROL 3 EVENT alarm NTROL 3 EVENT alarm	∃	
Actions:	Record video from BOSCH FLEX Resolution: 3072x1728 Recording rate: 15/s Pre-event recording: Off Post-event recording: On		-	
Pre-event recording time:	<u>0 s</u>			
Post-event recording time:	<u>10 s</u>			
Calendar:	The alarm is always enabled			
Special Days:				

4. Click **OK** to confirm alarm creation

Testing text channel alarm

- 1. Start Spotter
- 2. Click Alarms\Alarms and select Show



Sear File Alarms Sear				
view popup				
O Show				
Hide				
Auto Hide Cor				
1. Open used tex	t channel to	the work area		
127.0.0.15008				* X
Domo Mirasjs demo master server * Mirasjs demo master server	UDD4 DEMO			
Strice Front Door - UND-6070H				
RD Ansa - UND-6010R				
Office sade - XMD-LANBOR Office sade				
Extension Deput				
Test elem 2 INFUT				
Test skem 2 001P01 • TestChannels. •				
Mineys Slave server 1				
 Minanya stava server 2 * Source and the server 2 * 				
COUNTDOWNID COUNTDOWNID SCHEDULEDID				
Layout Physics				
Bookmarks Saved tabs	Time		Event	1

- 1. Download PuTTY
- 2. Install Putty and start the application
- 3. Set correct information
- 4. Click **Open**



Stepper:	n <u>a</u>	· ^
Category: - Session - Logging - Terminal - Keyboard - Bell - Features - Window - Appearance - Behaviour - Translation - Selection - Colours - Connection	Basic options for your PuTTY s	ession
	Specify the destination you want to conn Host Name (or IP address)	ect to Port 40000
	Connection type:	
	Load, save or delete a stored session Saved Sessions TextChannel Default Settings TextChannel	Load
- Proxy		Save
⊕-SSH Serial Telnet Rlogin SUPDUP		Delete
	Close window on exit: Always Never Only on	clean exit
		0.1

- 5. Type needed text to the PuTTY
- 6. Press Enter



Mirasys LtdC1CD, Vaisalantie 2-8, 02130Espoo, FinlandTel +358 (0)9 2533 3300-info@mirasys.com-www.mirasys.com-



- 1. Text channel shows received information
- 2. Event name is shown in the list
- 3. Alarm list show active alarm

Sameras X 🛞			Alarms		- *
10 22 47 8 2 2022		3	STREET ACCOUNTS OF THE REAL		
10.22.47 0.2.2022			Modern in the front door		
UDD4 DEMO			Mation in the office corri	and sectors in	
TCP Channel	Annual		Motion in the office corri	Tempore	
	Br DR-Master - Pully		Motion in the front door	2 minu	
	START ACCESS CONTROL 1 EVENT		Motion in the front door	2 mins	
			Monon in the office corri		
			Motion in the office corri		
			Motion in the office corri		
			Mation in the office corri		
START ACCESS CONTROL	1 EVENT				
START ACCESS CONTROL	1 EVENT alarm				
and the second			Motion in the front door		
			Motion in the office corri		
			Motion in the office corri		
			Motion in the front door		
			Motion in the office carri		
			Motion in the front door		
			Motion in the office carri		
			Motion in the frant door		
			Mution in the uffice carri	5 704705	
			Motion in the frunt door	Gimana	
			Motion in the office curry	C THEFT	
STATE STATE			Motion in the ritin data	T and the	
Time	Event		Motion in the time cont	7	
10 22 47 8 2 2022			Motion in the office coril	Timine	
10.22.47 0.2.2022	START ALLESS CONTROL T EVENT MAINT		Motion in the front door	-	
			Motion in the office corti		
			Mution in the office carri		
			Motion in the office corri		
			Motion in the front door		
			Motion in the front door		
			Show		

When the text channel receives ending input data, the data is shown in the text channel and alarm is ended

🏍 Cameras X 🕑		Alarms		¥×
10 24 28 8 2 2022		Motion in the office cord		-
10.24.28 8.2.2022		START ACCESS CONTROL	-2 mint	
UDD4 DEMO		Motion in the trong door	- 2 mars	
TCP Channel		Monton in the office core		
	B DE-Master - PUTTY - U X	Mentany in the office contri		
	START ACCESS CONTROL 1 EVENT	Motion in the front door	1 minut	
	STAL MOCESS COMING: 7 FEDE	Motion in the front door		
START ACCESS CONTROL	1 EVEN I			
START ACCESS CONTROL	1 EVENT alarm			
STOP ACCESS CONTROL 1	EVENT			
STOP ACCESS CONTROL 1	EVENT alarm	Motion in the front door		
STOP ACCESS CONTROL	EVENI didifi	Motion in the office cont	-1.000	
		Motion in the office cont.	- 5 mins	
		Motion in the troce door	20 mm	
		Musicon or the police cool		
		Motion in the trunt door	S mins	
		Motion in the office cont		
Time	Event	Motion in the front door		
Time	LYEIN	Motion in the office cont		
10.22.47 8.2.2022	START ACCESS CONTROL 1 EVENT alarm	Motion in the front door		
10.24.28 8.2.2022	STOP ACCESS CONTROL 1 EVENT alarm	Motion in the office corri		
		Matters in the from door		
1		Motion in the office cont	3 mins	
		Motion in the office corri		
		Motion in the office conti		
		Motion in the brant date	10 mins	
		Motion in the office coni		
		Motion in the office corri		
		Motion in the front door		
		Show		

Mirasys LtdC1CD, Vaisalantie 2-8, 02130Espoo, FinlandTel +358 (0)9 2533 3300-info@mirasys.com-www.mirasys.com



The last example shows that the text channel has received validated data

Second a second			Alarms		1.3
10.40.28 8.2.2022 UDD4 DEMO		×	START ACCESS CONTROL START ACCESS CONTROL Motion in the Trunt Boor	5 12 4 2 12 4 2 4 mins	
TCP Channel	BR-Master - PuTTV	- 0 X			
	START ACCESS CONTROL 1 EVENT		Motion in the office cont		
	STOP ACCESS CONTROL 1 EVENT		Motion in the office com		
	START ACCESS CONTROL 1 EVENT		Motion in the office curri	11 mm	
	START ACCESS CONTROL 3 EVENT		Motion in the office cont	13 mes	
			Motors in the office cord	14 month	
			START ACCESS CONTROL	18 mars	
START ACCESS CONTROL 1 EVENT					
START ACCESS CONTROL 1 EVENT alarm					
STOP ACCESS CONTROL 1 EVENT					
STOP ACCESS CONTROL & EVENT					
STOP ACCESS CONTROL 1 EVENT alarm					
START ACCESS CONTROL 1 EVENT			Mation in the front door		
START ACCESS CONTROL 1 EVENT alarm			Motion in the office corri-	20 mini	
START ACCESS CONTROL 2 EVENT			Motion in the office corri-	20 (9)(12)	
START ACCESS CONTROL 2 EVENT			Motion in the front door	20 minis	
START ACCESS CONTROL 2 EVENT alarm			Motion in the office corri		
START ACCESS CONTROL 3 EVENT					
START ACCESS CONTROL 3 EVENT alarm					
		416.4			
Time	Event				
10.22.47 8.2.2022	START ACCESS CONTROL 1 EVENT alarm		Motion in the front door		
10.24.28 8.2.2022	STOP ACCESS CONTROL 1 EVENT alarm		Motion in the office carri	22 mint	
10 40 15 9 2 2022	START ACCESS CONTROL 1 EVENT alarm		Motion in the Pont door	22 (1941)	
10.40.15 8.2.2022	START ACCESS CONTROL TEVENT diatrit		Motion in the front door	24 mini	
10.40.21 8.2.2022	START ACCESS CONTROL 2 EVENT alarm		Motion in the office corri	23 mini	
10.40.28 8.2.2022	START ACCESS CONTROL 3 EVENT alarm		Mation in the front door		



3. UDD4Demo.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
<logging>
<level value="2"/>
<additionalDebug value="no"/>
</logging>
<channelConfig>
<ignored value="0x00,0x0B"/>
<clearscreen value="----"/>
</channelConfig>
<validation>
<regex value=".*"/>
</validation>
<uddXmlMapper version="2">
<messageType value="text" parsing="regex">
<message number="1" value="alarm">
   <param number="1" value="([a-zA-Z]+) (.+)" group="1"/>
    <param number="2" value="([a-zA-Z]+) (.+)" group="2"/>
  </message>
</messageType>
 <constants>
   <array name="Rules" params="AT">
<value>start</value>
<value>stop</value>
</array>
   <array name="AlarmType" params="AT">
<value>temp</value>
      <value>normal</value>
<value>low</value>
</array>
</constants>
<rules>
    <message number="1" value="alarm" alwaysShowText="yes">
<param number="1" reference="1" value="$Rules" operator="eq" id="AT">
        <and number="1" reference="2" value="$AlarmType" operator="eq"/>
         <action number="1" type="event" value="$Rules $AlarmType alarm" />
</param>
   </message>
</rules>
</uddXmlMapper>
</root>
```

