DDS
Building Security Together

Amadeus 5 XML APIs

V 1.1

Publication No. 10UE572 Issue: March 28, 2010

TABLE OF CONTENTS

Introduction

This document is dedicated to explain the existing XML API of Amadeus5

It allows an **easy integration** with DDS Access Control and alarm monitoring software called Amadeus5.

This means that an external application could

- receive many information from Amadeus5 such as online events of access control system (Access granted, Access denied, Start of Alarm, …)
- act on the Access Control system by
    - Creating cardholders
    - Manages doors status, and relays status (Open a door for a while, open constantly, close constantly, or return to default status)
    - Manage alarm status (disarm a zone / input group)
    - Executing existing actions and processes of Amadeus5
    - Login / Logoff
    - User interface (messages on screen)
    - Download configuration to controllers (that may be updated directly in DB by an external application)

Amadeus5 also has other integration gateways such as

- OPC
- ModbusTCP
- Wizcon

Amadeus5 has existing integration with:

- Visitor management application (Telemaque www.safeware.fr)
- Windows account management (ISLOG www.islog.eu)
- LPR (Zamir)
- Outdoor Perimeter Security Systems (www.magal.co.il)
- Integration with RFID Readers on Pocket PC/PDA (External Events)
- Reception of ONSSI Video Systems alarms (External Events)

For more information, contact M. Nahir (info@dds-security.com)

The document is based on Amadeus5 Version 1.8.003 (June 2008)
Most of the commands are supported in previous versions, but in order to simplify we will only work on the basis of the actual version.
To get the latest version of Amadeus5 consult www.dds-security.com/am5

The communication with Amadeus5 is done by a communication engine called "Spread". For more information about Spread, see www.spread.org

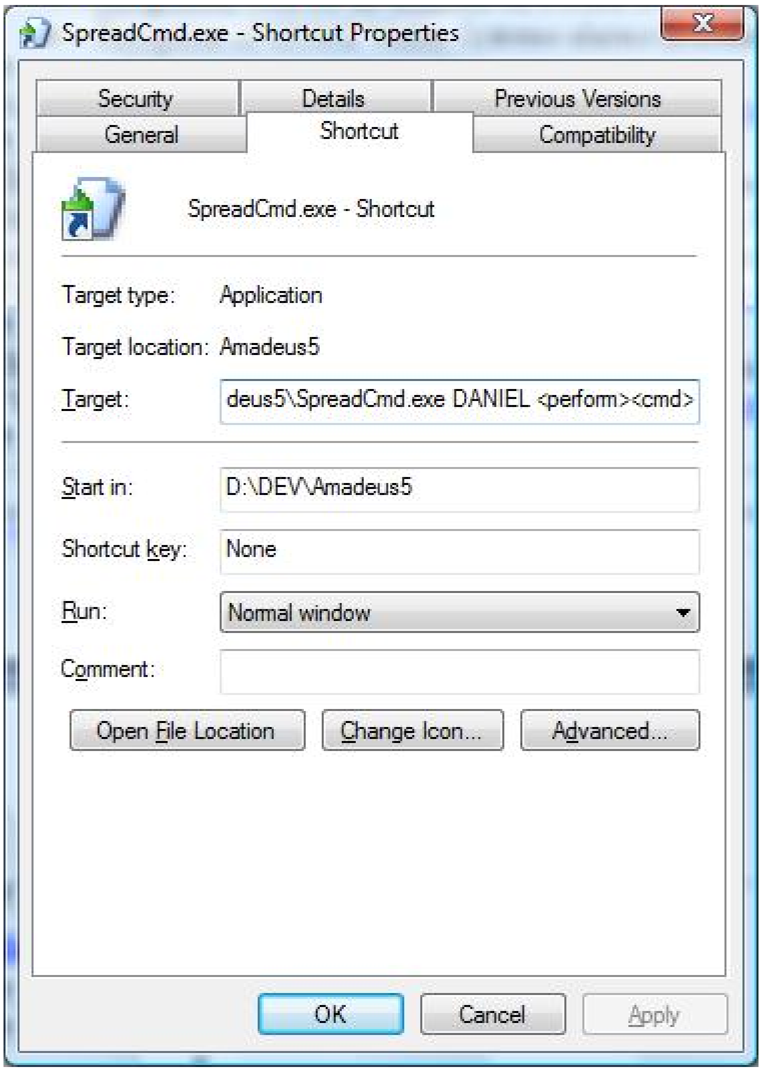
How to send/receive XML commands

Even before starting a software project, there are three methods to send XML commands to Amadeus5. The first two are 'one way', i.e., you can't see the answers, and the last method the answers are seen.

1. SpreadCmd (answers are not seen)

Target : D:\DEV\Amadeus5\SpreadCmd.exe DANIEL <perform><cmd>DisplayMessage</cmd><param><query><Param1> Hi, How are you? </Param1></query></param></perform>

Where DANIEL is the PC Name
And Amadeus5 should be running



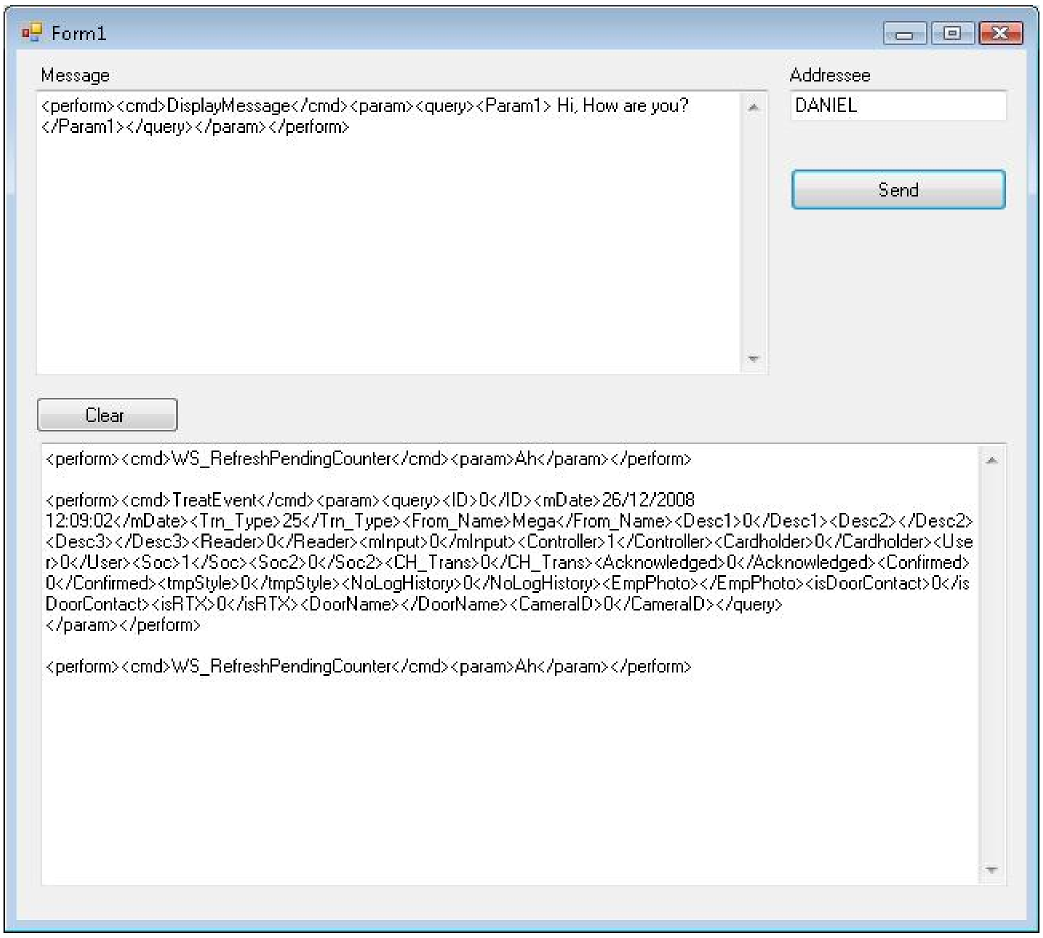
Target type: Application
Target location: Amadeus5
Target: deus5\SpreadCmd.exe DANIEL <perform><cmd>
Start in: D:\DEV\Amadeus5
Shortcut key: None
Run: Normal window
Comment:

2. Via a browser (answers are not seen)

See details on our publication:
'10TE570 Managing Amadeus5 Cardholders via HTTP'

3. XMLAPISample (EXE file)
(answers are seen on the lower part of the window)

See the project source code in VB.NET to see how to connect to Spread and send and receive XML message.
Note that Amadeus 5 should already be running.



Addressee: DANIEL

```
<perform><cmd>DisplayMessage</cmd><param><query><Param1> Hi, How are you?
</Param1></query></param></perform>
```

```
<perform><cmd>WS_RefreshPendingCounter</cmd><param>Ah</param></perform>

<perform><cmd>TreatEvent</cmd><param><query><ID>0</ID><mDate>26/12/2008
12:09:02</mDate><Trn_Type>25</Trn_Type><From_Name>Mega</From_Name><Desc1>0</Desc1><Desc2></Desc2>
<Desc3></Desc3><Reader>0</Reader><mInput>0</mInput><Controller>1</Controller><Cardholder>0</Cardholder><Use
r>0</User><Soc>1</Soc><Soc2>0</Soc2><CH_Trans>0</CH_Trans><Acknowledged>0</Acknowledged><Confirmed>
0</Confirmed><tmpStyle>0</tmpStyle><NoLogHistory>0</NoLogHistory><EmpPhoto></EmpPhoto><isDoorContact>0</is
DoorContact><isRTX>0</isRTX><DoorName></DoorName><CameraID>0</CameraID></query>
</param></perform>

<perform><cmd>WS_RefreshPendingCounter</cmd><param>Ah</param></perform>
```

The example uses a DLL (libtspd.dll) compiled for Windows platform.

For other platforms, please visit www.spread.org or specifically the page on supported platforms www.spread.org/SpreadPlatforms.html (Linux, Mac ...).
Note the spread version used by Amadeus5 is version 3.17.

Amadeus5 builds the spread.conf file and starts the spread daemon on its computer, from the definitions of PCs in Amadeus5 software.
If you need to work from another PC, there are 2 ways:

- Run a local daemon and connect to you local daemon (4803@localhost)

- Do not run any local daemon and connect to Amadeus5 PC daemon by connecting to 4803@<PCName or IP of Amadeus5 PC>

Important note:

Since version 1.8.206, there are 4 event groups:
-one for accesses
-one for I/O dynamic status
-one for debug info (mainly 'show commands')
-one for other events (denied, alarm, etc.)

If joining a group via the spread tool, external applications are able to listen to the messages of this group.

If you want, you can see these messages by using the sptuser.exe tool in the SpUtil folder of Amadeus5.
Launch Amadeus5 and then sptuser.
type 'j ag_1' for joining the group of accesses and then pass a valid card at the reader
you will see the corresponding message.

Here is the list of the groups:

ag_<Site ID> for Access granted
ot_<Site ID> for Other Events
dp_<Site ID> for Debug info
io_<ServerID> for IO dynamic status

Therefore, in the case of a 1 site (standard installation) the groups names are:

ag_1
ot_1
dp_1
io_1

XML Structure

All XML commands sent to/from Amadeus5 have the following structure:

```
<perform>
        <cmd>Name of the Command</cmd>
        <param>
                <query>
                        <Argument1>Value1</Argument1>
                        <Argument2>Value2</Argument2>
                        <Argument3>Value3</Argument3>
                        ...
                        <ArgumentN>ValueN</ArgumentN>
                </query>
        </param>
</perform>
```

The Cmd define the command name
The param contains a list of arguments

Methods

ChangeUserLogin

Send a request to change the user logged in by another one with the user name and password.

Syntax
Cmd: ChangeUserLogin
Parameters:
CmdLine=/us=UserName /pw=password
Where UserName and Password are the credential used in Login screen of Amadeus5.

Example:
```
<perform><cmd>ChangeUserLogin</cmd><param><query><CmdLine>/
us=dds /pw=dds</CmdLine></query></param></perform>
```

OpenScreen

Send a request to open a screen.
The command supports selecting

- on which record,
- on which tab
- and the screen size (Normal, minimize, or maximize).

Syntax
Cmd: OpenScreen
Parameters:
Param1 ScreenID (Cf Appendix A)
Param3= MinMax (0= Normal, 1=Minimized, 2 Maximized)
Param4= Only for Display Photo screen
onRecordID= on which record ID
onTabNumber= on which tab number (starting from 0)

Example:
```
<perform><cmd>OpenScreen</cmd><param><query><Param1>ID_Car
dholders</Param1><Param3>0</Param3><Param4>0</Param4><onRec
ordID>0</onRecordID><onTabNumber>0</onTabNumber></query></p
aram></perform>
```

Cf Appendix A (Screens ID) to get all the parameter according to the screen you want to open.

PreviewVideo

Send a request to preview a camera live video with the db_CameraID (cf Appendix B).

Syntax
Cmd: PreviewVideo
Parameters:

Param1 = db_CameraID (cf Appendix B)

Note that this command should be sent to the PC name (in uppercase) where we want the display video to be open.

Example:
```
<perform><cmd>PreviewVideo</cmd><param><query><Param1>1</Param1></query></param></perform>
```

PreviewReport

Send a request to preview an existing report with the report full name.

Syntax
Cmd: PreviewReport
Parameters:
Param1 = report full name with path

Example:
```
<perform><cmd>PreviewReport</cmd><param><query><Param1>D:\DEV\Amadeus5API\Reports\Last report.rpx</Param1></query></param></perform>
```

DisplayMessage

Send a request to display a message box with the text.

Syntax
Cmd: DisplayMessage
Parameters:
Param1 = message text

Example:
```
<perform><cmd>DisplayMessage</cmd><param><query><Param1> Hi, How are you? </Param1></query></param></perform>
```

MenuPrint (InsertTextinLog)

Send a request to insert message in the Log windows of Amadeus5.

Syntax:
Cmd: MenuPrint
Parameters:
st = message text
Soc = 1 by default. Use to filter information on Multi site / Multi company installation only
mStyle = Event Type. Will be display with the same color as defined for the event specified
JustinLog (Not used)

inViewPhotoAlso (Not used)

Example:
```
<perform><cmd>MenuPrint</cmd><param><query><st>Hi, How are
you ?
</st><Soc>1</Soc><mStyle>0</mStyle><JustinLog>0</JustinLog><in
ViewPhotoAlso>0</inViewPhotoAlso></query></param></perform>
```

FloodUpdateText (InsertTextinStatusBar)
Send a request to insert message in the status bar and set the percent of the progress bar.

Syntax:
Cmd: FloodUpdateText
Parameters:
pb = percentage number (0-100)
st = message text
srv (Not used)

Example:
```
<perform><cmd>FloodUpdateText</cmd><param><query><srv></srv>
<pb>50</pb><st>Hi, How are you ?
</st></query></param></perform>
```

PlaySound
Send a request to play a sound file with the full path of the sound file.

Syntax:
Cmd: PlaySound
Parameters:
Param1= sound file full name with path

Example:
```
<perform><cmd>PlaySound</cmd><param><query><Param1>C:\Wind
ows\Media\Windows Notify.wav</Param1></query></param></perform>
```

ExecuteAction
Send a request to execute an existing action with db_ActionID (cf Appendix B).

Syntax:
Cmd: ExecuteAction
Parameters:
pID = db_ActionID (cf Appendix B)

Example:
```
<perform><cmd>ExecuteAction</cmd><param><query><pID>1</pID>
</query></param></perform>
```

ExecuteProcess

Send a request to preview an existing process with db_ProcessID (cf Appendix B)

Syntax:
Cmd: ExecuteProcess
Parameters:
pID = db_ProcessID (cf Appendix B)

Example:
```
<perform><cmd>ExecuteProcess</cmd><param><query><pID>1</pID
></query></param></perform>
```

CC_RecreateMemoryTables (RecreateMemoryTables)

Send a request to initialize an existing controller db_ControllerID (cf Appendix B) with recreation of memory tables.

Syntax:
Cmd: CC_RecreateMemoryTables
Parameters:
CtrID = db_ControllerID (cf Appendix B)
ReStartPolling =
WantClearMemory =

Example:
```
<perform><cmd>CC_RecreateMemoryTables</cmd><param><query><C
trID>db_ControllerID</CtrID><ReStartPolling>0</ReStartPolling><Want
ClearMemory>1</WantClearMemory></query></param></perform>
```

ActivateRelay (SetRelayState)

Send a request to modify the relay state with the db_OutputID (cf Appendix B). Activate relay is only to activate a relay few seconds

Syntax:
Cmd: ActivateRelay
Parameters:
OutputID = db_OutputID (cf Appendix B)
Delay = x seconds (1 to 120 seconds)

Example: to activate relay ID 1 during 3 sec:
```
<perform><cmd>ActivateRelay</cmd><param><query><OutputID>1</
OutputID><Delay>3</Delay></query></param></perform>
```

OutputAction (SetRelayState)

Send a request to modify the relay state with the db_OutputID (cf Appendix B). OutputAction is used to set a relay state that remain permanently (until next change).

Syntax:
Cmd: OutputAction
Parameters:

OutputID = db_OutputID (cf Appendix B)
Action = action code
- 1 for normal mode
- 6 for constant on
- 7 for constant off

Example: to activate relay ID 1 constant on:
```
<perform><cmd>OutputAction</cmd><param><query><OutputID>1</
OutputID><Action>6</Action></query></param></perform>
```

Note this command update also the latest state of the relay in Amadeus5.

This command allows to control doors relays and other output (e.g. alarm siren)

ActivateInput (SetInputState)

Send a request to modify the Input state with the db_InputID (cf Appendix B). ActivateInput is used to pulse the input few seconds.

Syntax:
Cmd: ActivateInput
Parameters:

InputID = db_InputID (cf Appendix B)
Delay = multiple x 200ms (1 to 255 x 200 ms)

Example: to pulse input ID 1 during delay of 200 ms:
```
<perform><cmd>ActivateInput</cmd><param><query><InputID>1</In
putID><Delay>1</Delay></query></param></perform>
```

ActivateDeactiveInput (SetInputState)

Send a request to modify the Input state with the db_InputID (cf Appendix B). ActivateDeactiveInput is used to is used to set a input state that remain permanently (until next change).

Syntax:
Cmd: ActivateDeactiveInput
Parameters:

InputID = db_InputID (cf Appendix B)
CodeAction = action code
- 0 for Normal mode
- 8 for Deactivated

9 for Activated

Example: to deactivate input ID 1:
```
<perform><cmd>ActivateDeactiveInput</cmd><param><query><InputI
D>1</InputID><CodeAction>8</CodeAction></query></param></perfo
rm>
```

This command allows to control alarms sensors to be arm or not.

ActiveDeactiveInputGroup (SetInputGroupState)

Send a request to modify the Input group state (Deactivate / Force activate the input group) with the db_InputGroupID (cf Appendix B).

Syntax:
Cmd: ActiveDeactiveInputGroup
Parameters:

InputGroupID = db_InputGroupID (cf Appendix B)
InputGroupMode = action code

9 for Disarm during x seconds
10 for Disarm during x minutes
11 for Constant deactivated
12 for Return to normal mode (Cancel previous delay)
13 for Disarm until next time zone

14 for Activate during x seconds
15 for Activate during x minutes
16 for Constant activated
17 for Normal to normal mode (Cancel previous delay)
18 for Arm until next time zone

Delay = x seconds (1 to 60 seconds, 1-191 minutes)

Example: To disarm the input group 1 during 30 seconds
```
<perform><cmd>ActiveDeactiveInputGroup</cmd><param><query><In
putGroupID>1</InputGroupID><InputGroupMode>9</InputGroupMode>
<Delay>30/Delay></query></param></perform>
```

Example: To arm the input group 1 during 15 minutes
```
<perform><cmd>ActiveDeactiveInputGroup</cmd><param><query><In
putGroupID>1</InputGroupID><InputGroupMode>15</InputGroupMode
><Delay>10</Delay></query></param></perform>
```

This command allows to control alarm zones (defined as group of inputs) to be arm or not.

Methods with answer

GetTimeDate

Send a request to get the time and date of a controller db_ControllerID (cf Appendix B).

Syntax:
Cmd: GetTimeDate
Parameters:
CtrID = db_ControllerID (cf Appendix B)
SyncID = as the answer is return asynchronously, we define an number in the question that is returned in the answer to know the link between the answer and the question.
AnswerID contains in myID the group to whom the Amadeus5 server should answer. (We recommend to make such a group per PC)

Example:
Sent to Amadeus5:

```
<perform><cmd>GetTimeDate</cmd><param><query><CtrID>1</CtrI
D><SyncID>8</SyncID><AnswerID myID="API_DANIEL"
/></query></param></perform>
```

Response from Amadeus5:

```
<perform><cmd>syncGetResult</cmd><param><query><SyncID>8</S
yncID><Answer>16/12/2008
15:53:50</Answer></query></param></perform>
```

GetDigitalInputStatus

Send a request to get the input and output status of a controller db_ControllerID (cf Appendix B).
It returns the logical state of the input (physical state according to NO/NC) and the

Syntax:
Cmd: GetDigitalInputStatus
Parameters:
CtrID = db_ControllerID (cf Appendix B)
SyncID = see GetTimeDate explanation
AnswerID = see GetTimeDate explanation

The returned string is build of 0/1 in the following order

- Inputs (1-16)
- Relays (1-64)
- Inputs (17-24)
- Mega specific indication R1, R2, R3, R4, PSF, MS, 0, 0
  for Reader1 to 4 (R1-R4 : 1 if connected, 0 if not connected)
  PSF = Power Supply Failure input on board
  MS = MS input on board to indicate if box open or not.
  and two last values not used always 0

Example:
Sent to Amadeus5:
<perform><cmd>GetDigitalInputStatus</cmd><param><query><CtrID>1</CtrID><SyncID>3</SyncID><AnswerID myID="API_DANIEL" /></query></param></perform>

Response from Amadeus5:
<perform><cmd>syncGetResult</cmd><param><query><SyncID>3</SyncID><Answer>**000000000000000000000000000000000000000000000000000000000000000000000000000000000000000 - 11001100**</Answer></query></param></perform>

GetHardwareVersion

Send a request to get the hardware version of a controller db_ControllerID (cf Appendix B).
It returns a string. For more information, consult the TPL User Manual.

Syntax:
Cmd: GetHardwareVersion
Parameters:
CtrID = db_ControllerID (cf Appendix B)
SyncID = see GetTimeDate explanation
AnswerID = see GetTimeDate explanation

To understand the string return, please consult the TPL user manual.

Example:
Sent to Amadeus5:
<perform><cmd>GetHardwareVersion</cmd><param><query><CtrID>1</CtrID><SyncID>5</SyncID><AnswerID myID="API_DANIEL" /></query></param></perform>

Response from Amadeus5:
<perform><cmd>syncGetResult</cmd><param><query><SyncID>5</SyncID><Answer>**09 00 61 61 00 00 01 10 C7 07**</Answer></query></param></perform>

GetFirmareVersion

Send a request to get the firmware version of a controller db_ControllerID (cf Appendix B).
It returns the Eprom date and checksum.

Syntax:
Cmd: GetFirmareVersion
Parameters:
CtrID = db_ControllerID (cf Appendix B)
SyncID = see GetTimeDate explanation
AnswerID = see GetTimeDate explanation

The returned string is the date and check sum of the firmware.

Example:
Sent to Amadeus5:
```
<perform><cmd>GetFirmareVersion</cmd><param><query><CtrID>1<
/CtrID><SyncID>4</SyncID><AnswerID myID="API_DANIEL"
/></query></param></perform>
```

Response from Amadeus5:
```
<perform><cmd>syncGetResult</cmd><param><query><SyncID>4</S
yncID><Answer>10/07/08
3782</Answer></query></param></perform>
```

GetMemoryOccupation

Send a request to get the memory occupation of a controller db_ControllerID (cf Appendix B).
It returns the number of cardholders stored in the controller memory.

Syntax:
Cmd: GetMemoryOccupation
Parameters:

CtrID = db_ControllerID (cf Appendix B)
SyncID = see GetTimeDate explanation
AnswerID = see GetTimeDate explanation

Example:
Sent to Amadeus5:
```
<perform><cmd>GetMemoryOccupation</cmd><param><query><CtrID
>1</CtrID><SyncID>6</SyncID><AnswerID myID="API_DANIEL"
/></query></param></perform>
```

Response from Amadeus5:
```
<perform><cmd>syncGetResult</cmd><param><query><SyncID>6</S
yncID><Answer> 9</Answer></query></param></perform>
```

isPollingNow

Send a request to know if currently we are polling or not the controllers.
It returns True/False.

Syntax:
Cmd: GetMemoryOccupation
Parameters:

CtrID = db_ControllerID (cf Appendix B)
SyncID = see GetTimeDate explanation
AnswerID = see GetTimeDate explanation

Example:
Sent to Amadeus5:

<perform><cmd>isPollingNow</cmd><param><query><SyncID>7</SyncID><AnswerID myID="API_DANIEL" /></query></param></perform>

Response from Amadeus5:
<perform><cmd>syncGetResult</cmd><param><query><SyncID>7</SyncID><Answer>**1**</Answer></query></param></perform>

StartPolling

Send a request to Start Polling the controllers.
This command update the polling queues, it adds new controllers or remove controllers have been set as not active.
You can specify a specific controller or network. Without defining any controller (CtrID = 0), it starts the communication polling with all the controllers.

Syntax:
Cmd: StartPolling
Parameters:

CtrID = db_ControllerID (cf Appendix B)
NetID = db_NetwokID (cf Appendix B)
SyncID = see GetTimeDate explanation
AnswerID = see GetTimeDate explanation

Example:
```
<perform><cmd>StartPolling</cmd><param><query><CtrID>0</CtrID><NetID>0</NetID><SyncID>2</SyncID><AnswerID myID="API_DANIEL" /></query></param></perform>
```

Note that we send a SyncID even if we do not require a answer in order to force Amadeus5 to finish this request before processing another request.

StopPolling

Send a request to Stop Polling the controllers.
You can specify a specific controller. Without defining any controller (CtrID = 0), it stops all the communication polling with the controllers.

Syntax:
Cmd: StopPolling
Parameters:

CtrID = db_ControllerID (cf Appendix B)
SyncID = see GetTimeDate explanation
AnswerID = see GetTimeDate explanation

Example:
```
<perform><cmd>StopPolling</cmd><param><query><CtrID>0</CtrID><SyncID>3</SyncID><AnswerID myID="API_DANIEL" /></query></param></perform>
```

Note that we send a SyncID even if we do not require a answer in order to force Amadeus5 to finish this request before processing another request.

ImportCardholder

Send a request to import a cardholder in the database and inform the controllers. This allows adding, updating or deleting cardholders.

Syntax:
Cmd: ImportOneCardHolderXML
Parameters:

SyncID = see GetTimeDate explanation
AnswerID = see GetTimeDate explanation

Number =
Last_Name = name of the cardholder
First_Name = first name
Type
- 0 for Cardholder
- 1 for Visitor
- 2 for Guard
- 3 for Deleted

Badge = badge code (most of the time 8 digits)
Technology
- 1 for Magnetic
- 2 for BarCode
- 3 for Wiegand
- 4 for Wiegand2
- 5 for WiegandKeypad
- 6 for BioSmartCard
- 7 for Touch
- 8 for Radio

Photo = file name of the picture
Department = name of the department
Office_Phone
Access_Group= name of the access groups (separated with ;)
PIN_code = 4 digits
From_Date
To_Date
Validated = 1 for True, 0 for False
Street
City
ZIP
Personal_Phone
Description
Car_Number
ID
Supervisor
Label_1

Label_2
Label_3
Label_4
Company
Lift_Program = name of lift Program
Parking_Users_Group = name of Parking User Group
MultiSite_Type
    0 for Local
    1 for Shared
    2 for Global
Site = name of the site
Personal_WP = name of the Weekly Program
Personal_CL= value of the Crisis Level 0-7
Keep_card_on_motorized_reader = 1 for True, 0 for False
No_APB = 1 for True, 0 for False
No_access_during_holidays = 1 for True, 0 for False
Reset_APB = 1 for True, 0 for False
Need_Escort = 1 for True, 0 for False
Badge_Printing_Layout
Visited_person
Visited_person_location
Visit_purpose

Eye_Color = This field is an example of Customized Fields. They should be added in the XML in order to import them.

Example:
Sent to Amadeus5:
```
<perform><cmd>ImportOneCardHolderXML</cmd><param><query>
        <SyncID>17</SyncID>
        <AnswerID myID="API_DANIEL" />
        <Number>Dir784</Number>
        <Last_Name>Smith</Last_Name>
        <First_Name>John</First_Name>
        <Type>1</Type>
        <Badge>12345678</Badge>
        <Technology>3</Technology>
        <Photo></Photo>
        <Department></Department>
        <Office_Phone></Office_Phone>
        <Access_Group>Anytime Anywhere</Access_Group>
        <PIN_code></PIN_code>
        <From_Date>01/01/2008 08:00:00</From_Date>
        <To_Date></To_Date>
        <Validated>1</Validated>
        <Street></Street>
        <City></City>
        <ZIP></ZIP>
        <Personal_Phone>
        </Personal_Phone>
        <Description></Description>
        <Car_Number></Car_Number>
        <ID></ID>
        <Supervisor>1</Supervisor>
        <Label_1></Label_1>
        <Label_2></Label_2>
        <Label_3></Label_3>
        <Label_4></Label_4>
        <Company></Company>
        <Lift_Program></Lift_Program>
        <Parking_Users_Group></Parking_Users_Group>
        <MultiSite_Type>0</MultiSite_Type>
        <Site></Site>
        <Personal_WP></Personal_WP>
        <Personal_CL>0</Personal_CL>
        <Keep_card_on_motorized_reader>1</Keep_card_on_motorized_r
eader>
        <No_APB>1</No_APB>
        <No_access_during_holidays>1</No_access_during_holidays>
        <Reset_APB>1</Reset_APB>
        <Need_Escort>1</Need_Escort>
        <Badge_Printing_Layout></Badge_Printing_Layout>
        <Visited_person></Visited_person>
        <Visited_person_location></Visited_person_location>
        <Visit_purpose></Visit_purpose>
        <Eye_Color>Blue</Eye_Color>
</query></param></perform>
```

Response from Amadeus5:

```
<perform><cmd>syncGetResult</cmd><param><query><SyncID>17</
SyncID><Answer>0</Answer></query></param></perform>
```

The answer contains the result:

0 for UpdateSuccessfully
1 for InsertSuccessfully
10 for MandatoryFieldMissing
11 for UpdateFailed
12 for InsertFailed
13 for AuthorisationExcedded
14 for CannotChangeGuard
15 for DuplicateName
16 for CardHolderDeleted
17 for BadgeCodeNotOK

The import creates

- the cardholder,
- the badge,
- the access group if not found,
- the department if not found,
- the lift program if not found,
- the parking user group if not found,
- the personal weekly program if not found

It supports

- Multiple Access Group (use ; to separate the names of the access group)

- Dynamic Fields
- Multi site fields

For more details about the import, consult the user manual of Amadeus5 about import profiles.

Events

TreatEvent

Wake up the application when information to be displayed in Log windows arrives.

Syntax:
Syntax:
Cmd: TreatEvent
Parameters:

SyncID = see GetTimeDate explanation
AnswerID = see GetTimeDate explanation

Trn_Type = Transaction type (see Appendix C)

	Access granted	***Access denied***	***Unknown Card***	***Start of Alarm***	***End of Alarm***	***Technical Alarm***
ID	db_TableLOG_ID (cf Appendix B)					
mDate	Date of the event					
Trn_Type	1-2	3-4	61-63	10	11	22-29
From_Name	Reader name			Input Name		Controller name
Desc1	Transaction code		255	0 if immediate 1 if delayed	2	0
Desc2	Denied Reasons (see Appendix D)		0	Null		
Desc3	Cardholder name		Card code	Null		
EmpPhoto	Filename of the employee photo			"Bus2" if comes from Alarm priority bus		
NoLogHistory				1 if True, 0 if False		
isDoorContact				1 if True, 0 if False		
isRTX				1 if True, 0 if False		
DoorName				Reader name if RTX or Door contact		
Reader	db_ReaderID (cf Appendix B)]			0		
mInput	1 if escort else 0			db_InputID (cf Appendix B)		0
Controller	db_ControllerID (cf Appendix B)					
Cardholder	db_CardholderID (cf Appendix B)			0		0
User	0 in these case, db_UserID (cf Appendix B) for Login ..					
SOC	db_SocID (cf Appendix B)					
CH_Trans	Not used					
Acknowledged						

Confirmed	
tmpStyle	
Soc2	0 (useful only for multi company sites)
CameraID	db_CameraID (cf Appendix B)

Examples:

```
<perform><cmd>TreatEvent</cmd><param><query><ID>0</ID><mD
ate>16/12/2008
14:40:57</mDate><Trn_Type>1</Trn_Type><From_Name>Rdr01 /
Mega</From_Name><Desc1>0</Desc1><Desc2>0</Desc2><Desc3>Sm
ith John
00000003</Desc3><Reader>1</Reader><mInput>0</mInput><Controll
er>1</Controller><Cardholder>4</Cardholder><User>0</User><Soc>1
</Soc><Soc2>1</Soc2><CH_Trans>1</CH_Trans><Acknowledged>0</
Acknowledged><Confirmed>0</Confirmed><tmpStyle>0</tmpStyle><N
oLogHistory>0</NoLogHistory><EmpPhoto></EmpPhoto><isDoorContact
>0</isDoorContact><isRTX>0</isRTX><DoorName></DoorName><Cam
eraID>0</CameraID></query></param></perform>
```

16/12/08 14:40:57 Access Granted 'Smith John 00000003' From reader 'Rdr01 / Mega'

```
<perform><cmd>TreatEvent</cmd><param><query><ID>0</ID><mD
ate>16/12/2008
14:41:47</mDate><Trn_Type>3</Trn_Type><From_Name>Rdr02 /
Mega</From_Name><Desc1>0</Desc1><Desc2>8</Desc2><Desc3>Sm
ith John
00000003</Desc3><Reader>2</Reader><mInput>0</mInput><Controll
er>1</Controller><Cardholder>4</Cardholder><User>0</User><Soc>1
</Soc><Soc2>1</Soc2><CH_Trans>1</CH_Trans><Acknowledged>0</
Acknowledged><Confirmed>0</Confirmed><tmpStyle>0</tmpStyle><N
oLogHistory>0</NoLogHistory><EmpPhoto></EmpPhoto><isDoorContact
>0</isDoorContact><isRTX>0</isRTX><DoorName></DoorName><Cam
eraID>0</CameraID></query></param></perform>
```

16/12/08 14:41:47 Access Denied 'Smith John 00000003' From reader 'Rdr02 / Mega' - Not Authorized at this time

```
<perform><cmd>TreatEvent</cmd><param><query><ID>0</ID><mD
ate>16/12/2008
14:53:02</mDate><Trn_Type>11</Trn_Type><From_Name>i05 /
Mega</From_Name><Desc1>2</Desc1><Desc2></Desc2><Desc3></D
esc3><Reader>0</Reader><mInput>5</mInput><Controller>1</Contro
ller><Cardholder>0</Cardholder><User>0</User><Soc>1</Soc><Soc2
>0</Soc2><CH_Trans>0</CH_Trans><Acknowledged>0</Acknowledged
><Confirmed>0</Confirmed><tmpStyle>0</tmpStyle><NoLogHistory>0
</NoLogHistory><EmpPhoto></EmpPhoto><isDoorContact>0</isDoorCo
ntact><isRTX>0</isRTX><DoorName></DoorName><CameraID>0</Ca
meraID></query></param></perform>
```

16/12/08 14:53:02 End of alarm From input 'i05 / Mega'

```
<perform><cmd>TreatEvent</cmd><param><query><ID>0</ID><mD
ate>16/12/2008
14:53:03</mDate><Trn_Type>10</Trn_Type><From_Name>i01 /
Mega</From_Name><Desc1>0</Desc1><Desc2></Desc2><Desc3></D
```

esc3><Reader>0</Reader><mInput>1</mInput><Controller>1</Controller><Cardholder>0</Cardholder><User>0</User><Soc>1</Soc><Soc2>0</Soc2><CH_Trans>0</CH_Trans><Acknowledged>0</Acknowledged><Confirmed>0</Confirmed><tmpStyle>0</tmpStyle><NoLogHistory>0</NoLogHistory><EmpPhoto></EmpPhoto><isDoorContact>1</isDoorContact><isRTX>0</isRTX><DoorName>Rdr01 / Mega</DoorName><CameraID>0</CameraID></query></param></perform>

16/12/08 14:53:03 Start of Alarm From input 'i01 / Mega'- Immediate. Door forced 'Rdr01 / Mega'

ControllerCommunicationError

Wake up the application when a controller starts to be in Communication error.
It returns the text to be displayed in Log windows.
Example:
<perform><cmd>PollingError</cmd><param><query><Text>**16/12/2008 15:32:38 263 - Communication error Mega - Error 263. Timeout**</Text><CtrSoc>1</CtrSoc></query></param></perform>

ControllerCommunicationOK

Wake up the application when a controller returns to be in Communication OK.
It returns the text to be displayed in Log windows.

Example:
<perform><cmd>ComOK</cmd><param><query><Text>**16/12/2008 15:33:58 Communication OK Mega**</Text><CtrSoc>1</CtrSoc></query></param></perform>

Refresh_ioXML

Wake up the application when changes on input output status logical status arrives.
It returns the states of all input outputs of all the controllers.

Cf GetDigitalInputStatus methods for the format of the io string.

Example:
<perform><cmd>Refresh_ioXML</cmd><param><io server="DANIEL"><ctr1 io="00001000000000000000000000000000000000000000000000000000000000000000000000000000 - 11001100" server="DANIEL"/><ctr2 io="10000000000000000000000000000000000000000000000000000000000000000000000000000000 - 11001100" server="DANIEL"/></io></param></perform>

Contacting DDS for Technical Support

We recommend visiting our Web site to know the features of each product, and to be informed about new products.
http://www.dds-security.com

For Software Technical Support contact **Daniel ESSAYAG**

- By email help@dds-security.com
- By Phone 972-2-678-82-64/5
- By Fax 972-2-678-82-69

For Hardware Technical Support contact **Tsvika NAHIR**

- By email info@dds-security.com
- By Phone 972-2-678-82-64/5
- By Fax 972-2-678-82-69

For Sales information contact **Binyamin MACHADO**

- By email bin@dds-security.com
- By Phone 972-2-678-82-64/5
- By Fax 972-2-678-82-69

Appendix A: Screens ID

Screens ID	Description
ID_APBLevel	Anti Pass Back Level
ID_Area	Area
ID_Departement	Department
ID_Diagnostic	Diagnose
ID_Visitor	Visitor
ID_AccessGroup	Access Group
ID_Actions	Action
ID_Badge	Badge
ID_Cardholders	All Cardholders
ID_Computer	Computer
ID_Configuration	Customized Label
ID_Controllers	Controller
ID_Counters	Counter
ID_DailyProgram	Daily Program
ID_EventHandlingProgram	Event Handling Program
ID_GlobalReflex	Global Reflex
ID_InputGroup	Input Group
ID_OutputGroup	Output Group
ID_Holiday	Holiday
ID_Log	Active Alarms
ID_Network	Network
ID_Process	Process
ID_WeeklyProgram	Weekly Program
ID_ParkingDefinition	Parking Lot
ID_Company	Company / Site
ID_ZoneID	Parking User Group
ID_User	Users
ID_AuthorisationsLevels	Authorisation Levels
ID_Icons	Icons / Symbols
ID_Maps	Maps
ID_Positions	Position
ID_LiftAuthorisationGroups	Lift Authorisation group (only when Lift per Reader)
ID_LiftProgram	Lift program
ID_TimeAttendance	Roll Call
ID_CrisisLevel	Send a Crisis Level
ID_ExecuteProcess	Execute Process
ID_GuardDefinition	Guard Definition
ID_ViewPhoto	View Photo
ID_PatrolTour	Patrol Tour
ID_CheckPoint	Checkpoints
ID_PatrolStatus	Patrol status
ID_DisplayJournalSmall	Report wizard
ID_CreateagroupofBadges	Group of Badge
ID_ImportProfile	Import profiles
ID_CustomizedFields	Customized fields
ID_Camera	Camera
ID_Matrix	Matrix
ID_LocationStatus	Location Status

Appendix B: Database Fields

The database fields

db_ControllerID	Select ID, Name from Controller
db_ReaderID	Select ID, Name from Reader
db_InputID	Select ID, Name from [Input]
db_OuputID	Select ID, Name from [Output]
db_NetworkID	Select ID, Name from Network
db_SocID db_ReaderSocID db_CardHolderSocID	Select ID, Name from SOC
db_TableLOG_ID	Select ID from LOG
db_CardHolderID	Select ID, Last_Name & ‘ ‘ & First_Name as Name from CRDHLD
db_CameraID	Select ID, Name from Camera
db_InputGroupID	Select ID, Name from IGrp
db_ActionID	Select ID, Name from [Action]
db_ProcessID	Select ID, Name from Process

Appendix C: Transaction Type

The TRN_TYPE is describe in Param database in table Log_Events:

1	Access Granted
2	Access Granted + Duress code
3	Access Denied
4	Access Denied + unsuccessful successive trials
10	Start of Alarm
11	End of alarm
12	Line short
13	Line cut
14	Status 1 (Analog Input)
15	Status 2 (Analog Input)
16	Status 3 (Analog Input)
17	Status 4 (Analog Input)
22	Table Error
23	Low Battery
24	Power Down
25	Power Up
26	Power Supply Failure (input PSF closed)
27	Power Supply OK (input PSF opened)
28	Box Opened (input MS opened)
29	Box Closed (input MS closed)
31	Communication OK
32	Communication Error
33	Satellite alarm
3	Reader disconnected
35	Reader connected
40	User Acknowledgment
50	User Confirmation
51	User Comment
61	Unknown Card
62	Unknown card + unsuccessful successive trials
63	Non Allocated Badge
70	New record
71	Save record
72	Delete record
81	Application Login
82	Application Logout
90	Arrival
91	Early Arrival
92	No arrival on time
93	Late Arrival
94	Start guard tour
100	Scheduler
200	Initialize Controller

Appendix D: Denied Reasons

The denied reasons is decimal value that indicate a combination of 8 reasons

If the value = 250 means Supervisor
If the value = 255 means Access Group

For other values, the value should be change in binary.

1	- wrong finger
2	- Wrong Keypad Code
4	- Full / Lock / No answer from Door
8	- Not Authorized at this time
16	- Anti-Pass Back
32	- Reader not allowed
64	- Site Code not ok
128	- Inhibited Cardholder

If escort

1	- Card Unknown
2	- Wrong Keypad Code
4	- No card after 10 sec
8	- Not Authorized at this time
16	- Anti-Pass Back
32	- Inhibited Cardholder
64	- Site Code not ok
128	- Escort not authorized