Access Professional Edition

Personnel management

en User manual

Table of contents

1 Graphics

2 Copyright Information

License Agreement:
You should carefully read the following license agreement before proceeding with this installation. This Agreement is between you and Bosch Sicherheitssysteme GmbH, Postfach 1111, 85626 Grasbrunn ("Bosch"). By continuing with the installation, you indicate your acceptance of the terms of this legal Agreement. If you do not agree to the terms of this Agreement, promptly return this product to Bosch.

Copyright/Proprietary Protection:
This Bosch Access Professional Edition Software (the "Software") and the Documentation (all of the online help files and manuals, and all of the printed material included with this Software) are owned by Bosch and are protected by the German Law and international copyright laws and international treaty provisions. You must treat this software like any other copyrighted material, with the exceptions outlined in the following License Grant. Any violation of this agreement will automatically terminate your right to use this Software, and you must immediately return it to Bosch.

License Grant:
Bosch grants you a nonexclusive license to use this Software. You may copy this Software onto the hard disk of this computer, and make one copy for archival purposes. You may not make copies of the Software for any purpose other than what is stated above. You may not copy the Documentation for any reason. You may not reserve-engineer, disassemble, decompile or attempt to discover the source code of the Software. You may not modify the Software or create derivative products using this Software. You may not sublicense, rent or lease any portion of the Software. You may transfer your rights under this agreement on a permanent basis to another person or entity provided that you transfer this License Agreement, all original and updated Software and Documentation, and that you not retain any copies of the Software or Documentation. You must notify Bosch in writing of your transfer, and the recipient must also agree to the terms of this License Agreement.

No Warranties:
Bosch disclaims all warranties, either expressed or implied, including, but not limited to, implied warranties of merchantability and fitness for a particular purpose, with respect to this Software and the accompanying Documentation.

Limitation of Liability:
Under no circumstances, including negligence, shall Bosch, its employees, or its suppliers or resellers be liable for any direct, indirect, incidental, special, consequential, or any damages wathsoever (including, without limitation, damages for loss of profits, business interruption, attorneys fees, loss of information or other pecuniary loss) arising out of the use of or inability to use this Bosch product, even if Bosch has been advised of the possibility of such damages. Notwithstanding the foregoing, in no event shall the total liability of Bosch, or its employees, suppliers or resellers, for all damages, losses and causes of action, either jointly or severally, exceed the amount paid by you to Bosch or its resellers in the twelve (12) months prior to the claimed injury or damage.

Miscellaneous:
a. Nothing contained herein shall be deemed to convey to you any title or ownership interest in the Software or intellectual property rights related to such Software.
b. Any failure of Bosch to enforce any of the provisions of this Agreement will not be construed as a waiver of such right of Bosch to enforce each and every such provision.
c. If any provision of this Agreement shall be invalid, the invalid provision shall not affect the validity or enforceability of the remaining provisions of this Agreement.

d. This Agreement constitutes the entire agreement between you and Bosch and supersedes any prior agreement concerning the contents of the disc/diskette envelope. Bosch is not bound by any provision of any purchase order or any other type of correspondence (written or verbal).
e. This Agreement is governed by the laws of the Federal Republic of Germany

Use of the Sample Code:
Use of the Sample Code provided within the SDK for Cardholder data for the Bosch Access Professional Edition System:
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files, to deal with the Sample Code provided within the SDK included in the Software without restriction, including the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

THE SAMPLE CODE OF THE SOFTWARE DEVELOPMENT KIT IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Microsoft®, Windows® are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
Other product and company names mentioned herein may be the trademarks of their respective owners.

2.1 End User License Agreement

For countries in Americas see Agreement with Bosch Security Systems, Inc.
For all other countries see Agreement with Bosch Sicherheitssysteme GmbH
You accept the Agreement valid for your region.

2.1.1 Bosch Sicherheitssysteme GmbH

Bosch Security Systems B.V. ("Bosch") licenses this software and all associated documentation (the "Software") for nonexclusive use by the end user (herein called "Licensee"). Licensee has read this End User Software License Agreement (the "License") and agrees to abide by the terms and conditions of this License. By using the Software you, Licensee, accepts and agrees that Licensee will abide by, and are legally bound by, the terms of this License. If Licensee does not agree to abide by the terms of this License, Licensee shall immediately return the Software to Bosch. Licensee's use of the Software is subject to the following terms and conditions:
1. License. Under the terms of this nonexclusive, nontransferable (except as specifically permitted herein) License:
1.1 Licensee may use a machine-readable form of the Software on a single computer or a single server at a time and only for the operation of Bosch products.
1.2 Licensee may not modify, translate, create derivative works, decompile, disassemble, or reverse engineer the Software, except to the extent as permitted under applicable law, but in such case only for the purpose to enable interoperability of the Software with other systems.
1.3 Licensee may not sublicense, lease, or otherwise rent the Software without Bosch's prior written consent.

1.4 Licensee may make one copy of the Software solely for backup or archival purposes, provided such copy contains the original Software proprietary notice. No other copying of the Software is permitted.
1.5 This License will terminate automatically if Licensee fails at any time to comply with any of its terms or conditions.
1.6 Licensee may terminate this License at any time by returning the Software to Bosch and complying with the terms of Section 1.7 below.
1.7 Upon any termination of this License, Licensee shall immediately destroy the Software or return it to Bosch along with any copies Licensee has made, and delete any installed copy from Licensee's hardware. After termination of this License, Bosch will not provide any further support for the Software.
2. Transfer of Ownership. Licensee may transfer this Agreement and the License granted hereunder to another party only if Licensee:
2.1 also transfers the License, the Software, all accompanying documentation, and (by sale or lease) ownership of the associated Bosch hardware, if applicable,
2.2 requires the other party to abide by the terms of this License, and
2.3 destroys all copies of the Software, documentation and updates thereto that Licensee does not transfer to the other party.
3. Ownership and Propriety Rights. Although the diskette/media containing the Software is Licensee's, the Software is owned and copyrighted by Bosch and/or its suppliers. Except for the rights expressly granted herein, Bosch and its suppliers retain all rights to the Software, including, without limitation, the title to all copyright, patent, trade secret, and other intellectual and proprietary rights therein, and any copies thereof, in whole or in part, all of which are the valuable property of Bosch and/or its suppliers. Licensee may not remove, change, or delete the copyright notice from the Software. If Licensee makes any copies of the Software in whole or in part, all such copies shall contain the same copyright and proprietary markings as appear on or in the original Software copy, including diskette markings. Licensee will instruct its employees and others having access to the Software in, and ensure their compliance with the terms of, this License. Licensee shall use its best efforts to prevent any unauthorized copying of the Software. Licensee shall be responsible for any breach of any provision of this License by Licensee's employees. Licensee shall not sell, transfer, publish, disclose, or otherwise make available, the whole or any part of the Software, or any copies thereof, to any third party or persons not permitted by the terms of, and pursuant to the terms contained in this License. Licensee is not in violation of this Agreement, including this section, when a third party views the functional output resulting from Licensee's use of the Software.
4. Use of the Sample Code provided within the Software Development Kits (SDK) of the Bosch Video Management System, the Building Integration System or the Access Professional Edition:
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files, to deal with the Sample Code provided within the Software Development Kit (SDK) included in the Software without restriction, including the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
THE SAMPLE CODE OF THE SOFTWARE DEVELOPMENT KIT IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE

LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
5. Taxes. Licensee must pay all taxes that may now or hereafter be imposed, levied, or assessed, with respect to the possession or use of the Software or this License. Licensee shall file all reports required in connection with such taxes.
6. Taxes. The Licensee Fees for the Software is exclusive of all applicable federal, state, provincial and local taxes including, without limitation, sales, use, property, value added, goods and services, excise, and similar taxes, and all such taxes shall be assumed and paid by Licensee, excluding taxes on Bosch's net income. In the event that Bosch determines that any such taxes are subject to withholding requirements, Bosch may bill Licensee for such taxes, and Licensee shall promptly pay the amount billed. If any such tax for which Licensee is responsible hereunder is paid by Bosch, Licensee agrees to promptly reimburse Bosch therefore.
7. Warranty, Limitation of Liability, Remedies. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND INCLUDING WARRANTIES THAT THE SOFTWARE IS ERROR FREE OR WILL RUN UNINTERRUPTED, OR WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR AGAINST INFRINGEMENT. NEITHER BOSCH NOR ITS SUPPLIERS SHALL BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF USE, INTERRUPTION OF BUSINESS, LOSS OF DATA, LOSS OF PROGRAMMING AND/OR PRODUCTION MATERIALS, DAMAGE TO BUSINESS REPUTATION, OR FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY KIND, WHETHER UNDER THIS AGREEMENT OR OTHERWISE, OR FOR ANY CLAIM BY ANY OTHER PARTY. Bosch does not warrant the functions provided by the Software. However, Bosch warrants the diskette or other media on which the Software is furnished to be free from defects in materials and workmanship, under normal use, for a period of 90 days from the date of original purchase. Bosch's entire liability to Licensee, and Licensee's exclusive remedy, shall be the replacement of the diskette or other media not meeting Bosch's warranty, provided Licensee return the defective diskette or other media to Bosch. The replacement will be warranted for the remainder of the term of the original warranty, or 30 days, whichever is longer. Licensee is solely responsible for the selection of the Software to achieve Licensee's intended results, and for the installation, use, and results obtained from the Software. Without limiting the generality of the foregoing, in no event shall Bosch be liable for any indirect, incidental, special or consequential damages including, but not limited to, lost business, lost profits and other economic damages, whether foreseeable or not, even if advised of the possibility of such damages.
8. Export Control Laws.
8.1 Bosch's responsibility for delivery is limited to the delivery of the Software to Licensee. Bosch will not be responsible for obtaining any export licenses or re-export licenses which may be required for any subsequent shipping of the Software to destinations determined by Licensee.
8.2 In the event that Licensee, or any subsequent party handling the Software after delivery by Bosch to Licensee, does export the Software then, as between Bosch and Licensee:
8.2.1 Licensee shall be solely responsible for compliance with all applicable laws and regulations relating to such export including, but not limited to (a) export licenses or license exceptions, (b) determining correct classification at the time of export; and (c) any other regulatory agency approval requirements; and

8.2.2 Any diversion of the Software contrary to applicable law (including but not limited to U.S. law and the law of the jurisdiction in which Licensee is located) by Licensee or any such subsequent party is prohibited, and Licensee shall be solely responsible for any diversion which is contrary to applicable law.
8.2.3 Bosch may notify Licensee of any export issues identified by Bosch including, but not limited to, any export license requirements. Upon any such notification, the parties shall cooperate in good faith to achieve compliance with applicable laws and regulation relating to exports.
9. Term and Transfer. Licensee may terminate this License at any time by returning the Software to Bosch, or destroying the Software and all, together with all copies, in any form. Bosch may terminate this License if Licensee fails to comply with its terms and conditions in any material respect. Upon any termination, Licensee may not use the Software and must return or destroy all whole and partial copies thereof. After termination, Bosch will not further support the Software.
10. However for Open Source Software contained in this product the particular OSS license conditions have priority over the conditions of this EULA.
11. Governing Law. This Agreement shall be construed according to the laws of Switzerland. The provisions of the United Nations Convention on Contracts for the International Sale of Goods shall not apply. Any dispute will be subject to arbitration under the rules of the ICC and shall take place in Zurich, Switzerland.
12. Entire Agreement. This Agreement, any Bosch Standard Terms and Conditions as applicable from time to time, as well as all exhibits, schedules or appendices hereto, constitutes the complete and exclusive statement of the terms hereof and supersedes all prior oral and written statements of any kind made by the parties or their representatives with respect to the subject matter hereof. Any Customer purchase order or similar document issued by Customer shall not be part of this Agreement and shall not add to or modify any of the terms hereof. This Agreement may only be changed or supplemented by a written amendment signed by authorized representatives of the parties.

2.1.2 Bosch Security Systems, Inc.

Bosch Security Systems, Inc. ("Bosch") licenses this software and all associated documentation (the "Software") for nonexclusive use by the end user (herein called "Licensee"). Licensee has read this End User Software License Agreement (the "License") and agrees to abide by the terms and conditions of this License. By using the Software you, Licensee, accepts and agrees that Licensee will abide by, and are legally bound by, the terms of this License. If Licensee does not agree to abide by the terms of this License, Licensee shall immediately return the Software to Bosch. Licensee's use of the Software is subject to the following terms and conditions:
1. License. Under the terms of this nonexclusive, nontransferable (except as specifically permitted herein) License:
1.1 Licensee may use a machine-readable form of the Software on a single computer or a single server at a time and only for the operation of Bosch products.
1.2 Licensee may not modify, translate, create derivative works, decompile, disassemble, or reverse engineer the Software, except to the extent as permitted under applicable law, but in such case only for the purpose to enable interoperability of the Software with other systems.

1.3 Licensee may not sublicense, lease, or otherwise rent the Software without Bosch's prior written consent.
1.4 Licensee may make one copy of the Software solely for backup or archival purposes, provided such copy contains the original Software proprietary notice. No other copying of the Software is permitted.
1.5 This License will terminate automatically if Licensee fails at any time to comply with any of its terms or conditions.
1.6 Licensee may terminate this License at any time by returning the Software to Bosch and complying with the terms of Section 1.7 below.
1.7 Upon any termination of this License, Licensee shall immediately destroy the Software or return it to Bosch along with any copies Licensee has made, and delete any installed copy from Licensee's hardware. After termination of this License, Bosch will not provide any further support for the Software.
2. Transfer of Ownership. Licensee may transfer this Agreement and the License granted hereunder to another party only if Licensee:
2.1 also transfers the License, the Software, all accompanying documentation, and (by sale or lease) ownership of the associated Bosch hardware, if applicable,
2.2 requires the other party to abide by the terms of this License, and
2.3 destroys all copies of the Software, documentation and updates thereto that Licensee does not transfer to the other party.
3. Ownership and Propriety Rights. Although the diskette/media containing the Software is Licensee's, the Software is owned and copyrighted by Bosch and/or its suppliers. Except for the rights expressly granted herein, Bosch and its suppliers retain all rights to the Software, including, without limitation, the title to all copyright, patent, trade secret, and other intellectual and proprietary rights therein, and any copies thereof, in whole or in part, all of which are the valuable property of Bosch and/or its suppliers. Licensee may not remove, change, or delete the copyright notice from the Software. If Licensee makes any copies of the Software in whole or in part, all such copies shall contain the same copyright and proprietary markings as appear on or in the original Software copy, including diskette markings. Licensee will instruct its employees and others having access to the Software in, and ensure their compliance with the terms of, this License. Licensee shall use its best efforts to prevent any unauthorized copying of the Software. Licensee shall be responsible for any breach of any provision of this License by Licensee's employees. Licensee shall not sell, transfer, publish, disclose, or otherwise make available, the whole or any part of the Software, or any copies thereof, to any third party or persons not permitted by the terms of, and pursuant to the terms contained in this License. Licensee is not in violation of this Agreement, including this section, when a third party views the functional output resulting from Licensee's use of the Software.
4. Use of the Sample Code provided within the Software Development Kits (SDK) of the Bosch Video Management System, the Building Integration System or the Access Professional Edition:
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files, to deal with the Sample Code provided within the Software Development Kit (SDK) included in the Software without restriction, including the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

THE SAMPLE CODE OF THE SOFTWARE DEVELOPMENT KIT IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

5. Taxes. Licensee must pay all taxes that may now or hereafter be imposed, levied, or assessed, with respect to the possession or use of the Software or this License. Licensee shall file all reports required in connection with such taxes.

6. Taxes. The Licensee Fees for the Software is exclusive of all applicable federal, state, provincial and local taxes including, without limitation, sales, use, property, value added, goods and services, excise, and similar taxes, and all such taxes shall be assumed and paid by Licensee, excluding taxes on Bosch's net income. In the event that Bosch determines that any such taxes are subject to withholding requirements, Bosch may bill Licensee for such taxes, and Licensee shall promptly pay the amount billed. If any such tax for which Licensee is responsible hereunder is paid by Bosch, Licensee agrees to promptly reimburse Bosch therefore.

7. Warranty, Limitation of Liability, Remedies. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND INCLUDING WARRANTIES THAT THE SOFTWARE IS ERROR FREE OR WILL RUN UNINTERRUPTED, OR WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR AGAINST INFRINGEMENT. NEITHER BOSCH NOR ITS SUPPLIERS SHALL BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF USE, INTERRUPTION OF BUSINESS, LOSS OF DATA, LOSS OF PROGRAMMING AND/OR PRODUCTION MATERIALS, DAMAGE TO BUSINESS REPUTATION, OR FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY KIND, WHETHER UNDER THIS AGREEMENT OR OTHERWISE, OR FOR ANY CLAIM BY ANY OTHER PARTY. Bosch does not warrant the functions provided by the Software. However, Bosch warrants the diskette or other media on which the Software is furnished to be free from defects in materials and workmanship, under normal use, for a period of 90 days from the date of original purchase. Bosch's entire liability to Licensee, and Licensee's exclusive remedy, shall be the replacement of the diskette or other media not meeting Bosch's warranty, provided Licensee return the defective diskette or other media to Bosch. The replacement will be warranted for the remainder of the term of the original warranty, or 30 days, whichever is longer. Licensee is solely responsible for the selection of the Software to achieve Licensee's intended results, and for the installation, use, and results obtained from the Software. Without limiting the generality of the foregoing, in no event shall Bosch be liable for any indirect, incidental, special or consequential damages including, but not limited to, lost business, lost profits and other economic damages, whether foreseeable or not, even if advised of the possibility of such damages.

8. U.S. Government Restricted Rights. The Software is provided with restricted rights. Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFAR 252.227-7013, Federal Acquisition Regulation clause 52.227-19 (c) (2) Commercial Computer Software Restricted Rights, NASA clause 52.227.86 (d) Commercial Computer Software Licensing, or their successor.

9. Export Control Laws.

9.1 Bosch's responsibility for delivery is limited to the delivery of the Software to Licensee. Bosch will not be responsible for obtaining any export licenses or re-export licenses which may be required for any subsequent shipping of the Software to destinations determined by Licensee.
9.2 In the event that Licensee, or any subsequent party handling the Software after delivery by Bosch to Licensee, does export the Software then, as between Bosch and Licensee:
9.2.1 Licensee shall be solely responsible for compliance with all applicable laws and regulations relating to such export including, but not limited to (a) export licenses or license exceptions, (b) determining correct classification at the time of export; and (c) any other regulatory agency approval requirements; and
9.2.2 any diversion of the Software contrary to applicable law (including but not limited to U.S. law and the law of the jurisdiction in which Licensee is located) by Licensee or any such subsequent party is prohibited, and Licensee shall be solely responsible for any diversion which is contrary to applicable law.
9.2.3 Bosch may notify Licensee of any export issues identified by Bosch including, but not limited to, any export license requirements. Upon any such notification, the parties shall cooperate in good faith to achieve compliance with applicable laws and regulation relating to exports.
10. However for Open Source Software contained in this product the particular OSS license conditions have priority over the conditions of this EULA.
11. Term and Transfer. Licensee may terminate this License at any time by returning the Software to Bosch, or destroying the Software and all, together with all copies, in any form. Bosch may terminate this License if Licensee fails to comply with its terms and conditions in any material respect. Upon any termination, Licensee may not use the Software and must return or destroy all whole and partial copies thereof. After termination, Bosch will not further support the Software.
12. Governing Law. This Agreement shall be construed according to the laws of the State of New York and the U.S. The provisions of the United Nations Convention on Contracts for the International Sale of Goods shall not apply. Any dispute will be subject to arbitration under the rules of the American Arbitration Association and shall take place in the Metropolitan area of Rochester, New York, U.S.A.
13. Entire Agreement. This Agreement, any Bosch Standard Terms and Conditions as applicable from time to time, as well as all exhibits, schedules or appendices hereto, constitutes the complete and exclusive statement of the terms hereof and supersedes all prior oral and written statements of any kind made by the parties or their representatives with respect to the subject matter hereof. Any Customer purchase order or similar document issued by Customer shall not be part of this Agreement and shall not add to or modify any of the terms hereof. This Agreement may only be changed or supplemented by a written amendment signed by authorized representatives of the parties.

3 Introduction

The Access Professional Edition SDK can be used to help you integrate access control functionalities into your applications based on the Access Professional Edition version 3.5 or later.

What is in this guide?

This guide provides detailed descriptions of the SDKs that can be used to develop access control applications.

Who should read this guide?

This guide is meant for C# developers who would like to develop access control applications. The SDK can be used with .NET framework 4.

The demo app as part of the distribution package can be modified with Microsoft® Visual Studio 2013.

4 Overview

The Access Professional Edition Software Development Kit (SDK) can be used by developers to integrate access control functionalities into their applications. The SDK can be used with Access Professional Edition version 3.5 or later and provides the following functions to 3rd party applications:

- Read all person and card data
- Create, update and delete a person
- Assign and delete card for the specified person
- Read all configured entrances
- Assign authorization of an entrance to a person/groups
- Read any entry and existing transaction of a person
- Event trigger for entries and transactions
- Read all entries and existing transactions filtered by time frame
- Direct control of the access to doors
- Read door and operation state of configured entries.
- Event trigger for door or operation state changes of entrances
- Event trigger for person changes
- Event trigger for threat alert
- Read areas with person count and state

This document describes the use of the SDK for manipulating Access Professional Edition Cardholder Data from 3rd party applications.

4.1 Fundamentals

The Access Professional Edition SDK consists of methods that are used to manage the objects of the Access Professional Edition database. The SDK comes in the form of standard libraries, where methods and object classes are operated within C# programming environments.
For C# applications, a set of interop wrappers is provided, namely Bosch_APE_SDK_CS.dll.
The Interface-DLL "BOSCH_APE_SDK_CS.dll" uses the following additional DLLs internally:

- `bcl-1.dll`
- `Bosch_APE_SDK.dll`
- `Cdf-2.dll`
- `csl-2.dll`
- `cximageMfc.dll`
- `DbiCtree.dll`
- `slodbc-2.dll`
- `IpcClass2.dll`
- `NetSec-2.dll`
- `Pegasys.dll`
- `ms.dll`
- `UserRights-1.dll`

These DLLs are located in the binary folder of the installation. It is recommended to copy these DLLs into your own development environment.

4.2 What is new in Version 1.5

- Version 1.5 contains functions for handling threat alerts.

Notice!
There was a change to the assignment of access authorizations in SDK-Version 1.4. If you are upgrading from a version previous to 1.4, and are using access authorizations, you will need to adapt your programs to the structure used in SDK-Version 1.4 (see below).

5 Installation

When installing the Access Professional Edition SDK, the documentation of the interfaces is included along with the application. The document is in the help format.

Supported platforms are:
- Windows 7 SP1 (32-bit, 64-bit in 32-bit emulation mode), .NET framework 4 required.
- Windows 10 (64-bit), .NET framework 4 required.

All Microsoft® updates and hotfixes are expected to be installed on target PCs. Graphics card drivers are also expected to have the latest officially released version.

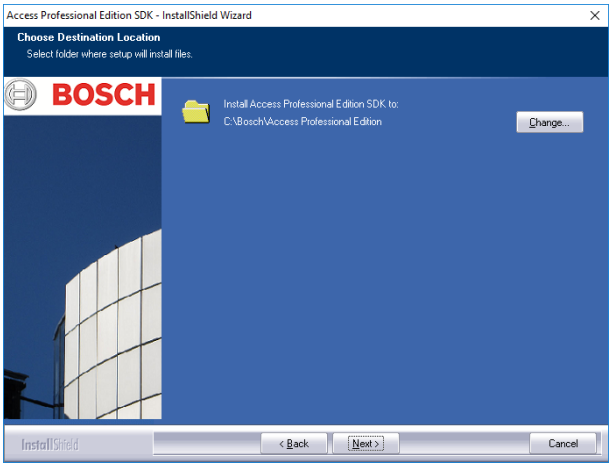
The **Bosch APE_SDK_CS.DLL** file is used by the developer who is using C# as the development language. This DLL contains several classes, header files and methods. The main class for this DLL is **APE_SDK** which contains method where the 3rd party applications can be used to perform operations in the Access Professional Edition.

5.1 Access Professional Edition SDK

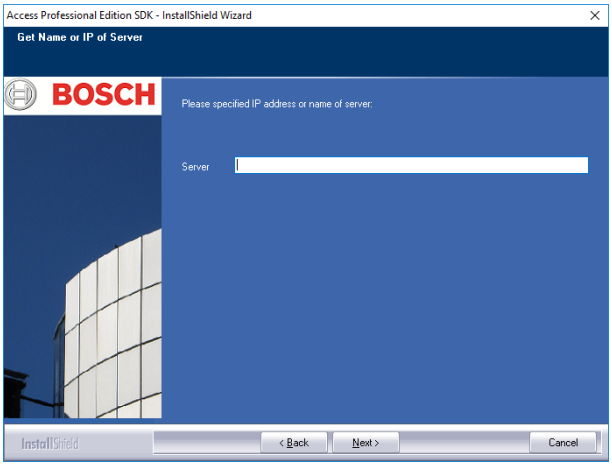
Make sure that you have at least:
- an **APE 3.2 basic license,**
- an **ASL-APE3P-SDK license**

If these preconditions apply, start the **APE Configurator > Configuration > License activation**.
Make sure that the **Access Professional Edition SDK** is active on your APE system.
The SDK for cardholder data can be installed :
- On the same PC, where the APE server is installed
- On the same PC, where the APE client is installed
- On a stand-alone PC, no APE client or server is installed on this machine.

Run the process until it is completed.



If an Access Professional Edition server or client is already installed on the PC, choose the recommended folder on the same drive.
During Installation, you are asked to enter the PC name or IP address of the computer, which hosts the APE server.



For a functional verification of the installation, start the demo app and connect to the **Access Professional Edition Server**.

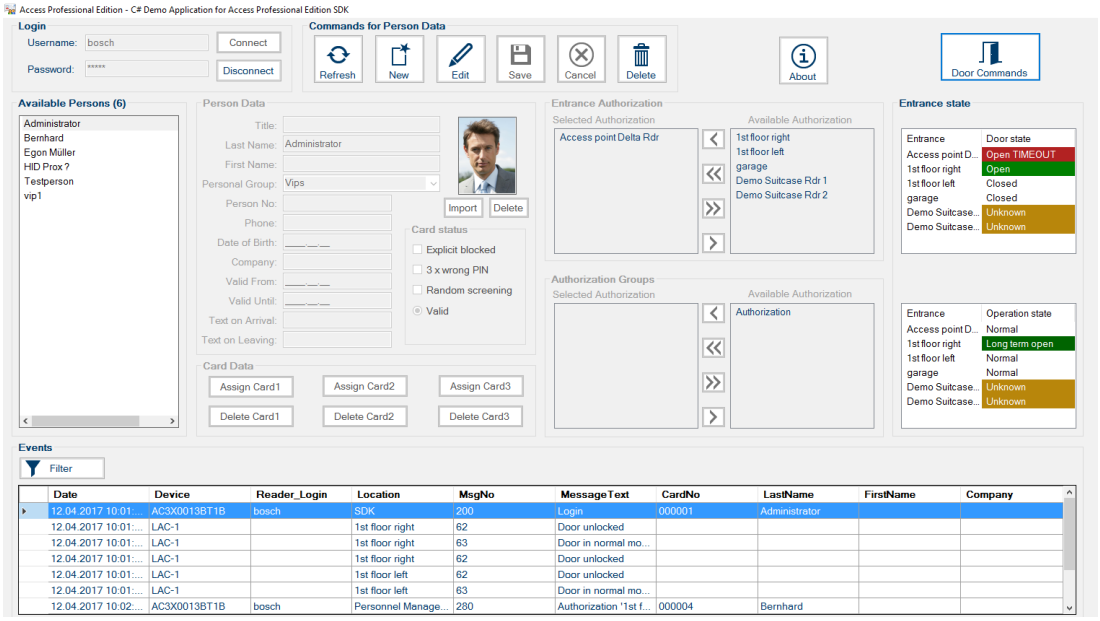
5.2 Version Information

Caution!
Access Professional Edition SDK and the Access Professional Edition work properly together only as long as both are compatible.
For details refer to the release notes of the Access Professional Edition.
Bevor installing the latest Access Professional Edition SDK version, make sure that previous versions are uninstalled.

5.3 Demo Application

A demo application is available as an example for the usage of the Access Professional Edition SDK. The demo application is a simple application with one main screen to demonstrate all available SDK methods.

Notice!
The Demo Application is intended for demonstration purpose only!
It is not designed to replace the Access Professional Edition Client in a productive environment.



The following transactions can be performed in the **Demo Application**:

- Create a person in the Access Professional Edition
- Read all the person from the Access Professional Edition
- Read detailed information of a specific person
- Delete a person except the administrator user
- Get the available authorizations [entrances]
- Get a new card number which can be used while creating or updating a person
- Get information on the entry and exit of a person
- Handle Open-/Close door commands
- Get door and operation state of entrances
- Event trigger for door or operation state changes
- Event trigger for person changes
- Event trigger for threat alert
- Event trigger for messages
- Event trigger for configuration changes
- Set / Reset threat alert
- Assign functional card for threat alert
- Read areas with person count and state

6 List of Classes

The following section lists all the classes and their descriptions.

6.1 Classes of Persons

6.1.1 APE_Assigned Authorization Class

A list of this object is used by the C# developer for the time restricted assignment of authorizations to a person in the APE_PERSON class.
This list replaces the list of assigned authorizations in the APE_Person class in version 1.3.

Member Variables	Description
AuthNo	Authorization Number [integer]
ValidFrom	Authorization valid from with yyyymmddhhmmssz format [string]
ValidTo	Authorization valid expiry with yyyymmddhhmmssz format [string]

6.1.2 APE_Assigned Authorization Group Class

A list of this object is used by the C# developer for the time restricted assignment of authorization groups to a person in the APE_PERSON class.
This list replaces the list of assigned authorization groups in the APE_Person class in version 1.3.

Member Variables	Description
GroupNo	Authorization Group Number [integer]
ValidFrom	Authorization valid from with yyyymmddhhmmssz format [string]
ValidTo	Authorization valid expiry with yyyymmddhhmmssz format [string]

6.1.3 APE_Person Class

This object is used by C# developer for detailed information about a person. The object is used on Add, Change or Delete a person.

Member Variables	Description
Person ID	Person ID [integer]
PersNo	Person Number [integer]
Valid	Person status [integer]
PersonalGroup	PersonGroup ID [integer]
LastName	Person last name [string]
FirstName	Person first name [string]
Title	Person title [string]
DisplayName	Person display name [string]

Member Variables	Description
Name	Person Name [string]
Company	Company information [string]
HandyNo	Person mobile number [string]
DateOfBirth	Date of birth of a person with format yyyymmdd [string]
Pin	Verification pin with 4 digits [string]
TextComing	Text to be displayed in reader on arrival [string]
TextLeaving	Text to be displayed in reader on leaving [string]
PictureFullFileName	Overwrite this string with a full file specification to a local picture (*.jpg) for importing a person's picture [string]
PictureFileName	Person picture file name [string]
CardValidFrom	Card valid from with format yyyymmdd [string]
CardValidUntil	Card expiry date with format yyyymmdd [string]
Remarks	Remarks about the person [string]
Memo	Person memo [string]
CardNo1	First card number [integer]
CodeNo1	First card code [string]
CardVersion1	First card version [integer]
CodeNoVersion1	First card code version [integer]
CountryCode1	First card country code [integer]
CustomerCode1	First card customer code [integer]
CardNo2	Second card number [integer]
CodeNo2	Second card code [string]
CardVersion2	Second card version [integer]
CodeNoVersion2	Second card code version [integer]
CountryCode2	Second card country code [integer]
CustomerCode2	Second card customer code [integer]
CardNo3	Third card number [integer]
CodeNo3	Third card code [string]
CardVersion3	Third card version [integer]
CodeNoVersion3	Third card code version [integer]
CountryCode3	Third card country code [integer]
CustomerCode3	Third card customer code [integer]
AreaId	Person location [integer]

Member Variables	Description
AssignedAuthorizationList	List of APE_AssignedAuthorization
AssignedAuthGroupList	List of APE_AuthorizationGroup

6.2 Authorization Class

This class holds the members of authorization.

6.2.1 APE_Authorization Class

This object is used by the developer who is using C# as the development language.

Member Variables	Description
AuthNo	Authorization Number [integer]
ValidFrom	Authorization valid from with yyyymmdd format [string] (not yet used, for future use)
ValidTo	Authorization valid expiry with yyyymmdd format [string] (not yet used, for future use)
AuthName	Authorization name [string]
Type	Authorization type [string]

6.3 AuthorizationGroup Class

This class holds the members of authorization groups.

6.3.1 APE_AuthGroup Class

This object is used by the developer who is using C# as the development language.

Member Variables	Description
GroupNo	Authorization group Number [integer]
ValidFrom	Authorization valid from with yyyymmdd format [string] (not yet used, for future use)
ValidTo	Authorization valid expiry with yyyymmdd format [string] (not yet used, for future use)
AuthName	Authorization group name [string]

6.4 PersonGroup Class

This class holds the details of person group.

6.4.1 APE_PersonGroup Class

This object is used by the developer who is using C# as the development language.

Member Variables	Description
PersonGroupID	Person Group ID [integer]
PersonGroupName	Person Group Name [string]
PersonGroupDescription	Person Group Description [string]
Active	Group status [integer]

6.5 Entrances Class

This class holds the details of entrances.

6.5.1 APE_Entrances Class

This object is used by the developer who is using C# as the development language.

Member Variables	Description
No	Entrance Number [integer]
Name	Entrance Name [string]

6.6 Message Class

This class holds the details of messages.

6.6.1 APE_Message Class

This object is used by the developer who is using C# as the development language.

Member Variables	Description
No	Message Number [integer]
Name	Message Description [string]

6.7 Event Class

This class holds the details of single event.

6.7.1 APE_Event Class

This object is used by the developer who is using C# as the development language.

Member Variables	Description
LACNo	LAC Number [integer]
GID	GID used in AC config entrance dialog [integer]
DID	Device ID [integer]
DoorNo	Door number [integer]
DestNo	Destination [integer]
Date	Event date and time [string]
MsgNo	Message number [integer]
CodeNo	Card code for any card related events [string]
ExtraData	Extra event information [string]
Device	Device name in which the event is generated [string]
Reader	Reader name in which the event is generated [string]
Cat	Event category normal event/ alarm[integer]
MessageText	Event description [string]
PersonID	Person ID who’s involved in event [string]
CardNo	Card number who’s involved in event [string]
PersonNo	Person number who’s involved in event [string]
LastName	Person last name who’s involved in event [string]
FirstName	Person first name who’s involved in event [string]
Company	Company ID [string]
MessageType	Event type [string]
AdditionalInfo	Additional information about the event [string]
DateLAC	Date in LAC [string]
AlarmCam	Alarm camera ID [integer]
CamImage	Surveillance camera ID [integer]

6.8 EventFilter Class

This class holds the details of events filter.

6.8.1 APE_EventFilter Class

This object is used by the developer who is using C# as the development language.

Member Variables	Description
LastName	Lastname of person [integer]
Company	Company name [string]
Messages	Message ID, more than one message can be assigned using ';' separated [string]
Entrances	Entrance ID, more than one entrance can be assigned using ';' separated [string]
CardNo	Card number [string]
FromDate	From date [string]
ToDate	To date [string]

6.9 Entrance State Class

6.9.1 APE_eEntranceState enum Class

enum Entrance State	Description
ENTRANCE_STATE_DEFECT = 0	Door state DEFECT
ENTRANCE_STATE_UNKNOWN = 1	Door state UNKNOWN
ENTRANCE_STATE_OK = 2	Door state OK (closed)
ENTRANCE_STATE_OPEN = 3	Door state OPEN
ENTRANCE_STATE_BREAK = 4	Door state BREAK
ENTRANCE_STATE_OPEN_TIMEOUT = 5	Door state OPEN-TIMEOUT (open too long after access)
ENTRANCE_STATE_OPEN_PERMANENT = 6	Operation state OPEN PERMANENT
ENTRANCE_STATE_LOCK_PERMANENT = 7	Operation state LOCK-PERMANENT

6.9.2 APE_EntranceState

Member Variables	Description
No	Entrance Number [integer]
Name	Entrance Name [string]
DoorState	Door State [APE_eEntranceState]
Operation	Operation State [APE_eEntranceState]

6.10 Person Change class

This class holds the details of a person change event.

6.10.1 APE_ePersonAction enum Class

enum Persons Action	Description
PERSON_TA_EVENT = 0	Time attendent event (entry / leave)
PERSON_VALID_CHANGED = 1	Persons valid changed (locked etc.)
PERSON_DATA_CHANGED = 2	Any persons data changed
PERSON_DELETED = 3	Person deleted

6.10.2 APE_PersonChange Class

Member Variables	Description
Action	Action [APE_ePersonAction]
PersonID	Person-ID [integer]
apePerson	Persons data [APE_Person]

Notice!
The Member **apePerson** (APE_Person Class) is not valid on delete. Use member PersonID to serach for deleted person in your data lists.

7 List of Methods

The following section lists all the methods that are applicable for the classes.

7.1 Connect()

Description:
Use this method to create an instance to this class and get connected to the Access Professional Edition server.

Parameters:

C#	`Username`	Valid APE username.
	`Password`	Valid APE password.

Definition:

C#	`int Connect(String^ Username,String^ Password);`

Example:

C#	`APE_SDK ac = new APE_SDK();` `int ret = ac.Connect("abc","xyz");`

7.2 Disconnect()

Description:
Use this method to dispose objects and disconnect from Access Professional Edition. This method must be called while closing the application or during disconnect.

Parameters:
None.

Definition:

C#	`int Disconnect();`

Example:

C#	`APE_SDK ac = new APE_SDK();` `m_ac.Disconnect();`

7.3 ReadAllPerson()

Description:
Use this method to get all available person information. The person list will be filled on the parameter.

Parameters:

C#	`APE_Person object list`

Definition:

C#	`int ReadAllPerson(List<APE_Person^>^ m_AcPerlist);`

Example:

C#	`List<APE_Person> personlist = new List<APE_Person>();` `int val = ac.ReadAllPerson(personlist);`

7.4 GetPerson()

Description:
Use this method to get detailed information of a specific person.

Parameters:

C#	`PersonID`	Required Person ID
	`APE_Person object`	This method fills the person details.

Definition:

C#	`int GetPerson(int PersonID, APE_Person^ AcPerson);`

Example:

C#	`APE_Person person = new APE_Person();` `int ret = ac.GetPerson(1, person);`

7.5 CreatePerson()

Description:
Use this method to create a person with specified information. Before calling this method, all required information has to be filled in the Person object.

Parameters:

C#	`APE_Person object`	Person object with details.

Definition:

C#	`int CreatePerson(APE_Person^ APEPerson);`

Example:

C#	`APE_Person p;`
	`p.title = "Mr";`
	`p.lastName = "Person";`

```
p.firstName =”Test”;
p.personalGroup = 1;
int ret = ac.CreatePerson(p);
```

7.6 UpdatePerson()

Description:
Use this method to update a person with specified information. Before calling this method, any modified information has to be filled in the existing Person object.

Parameters:

C#	`APE_Person object`	Person object with details.

Definition:

C#
```
int UpdatePerson(APE_Person AcPerson);
```

Example:

C#
```
APE_Person pers;
pers.title = “Mr”;
pers.lastName = “Person1”;
pers.firstName =”Test1”;
pers.personalGroup = 1;
int ret = ac.UpdatePerson(pers);
```

7.7 DeletePerson()

Description:
Use this method to delete a specified person.

Parameters:

C#	`APE PersonID`	Person ID to be deleted.

Definition:

C#
```
int DeletePerson(int PersonID);
or
int DeletePerson(APE_Person^ APEPerson);
```

Example:

C#
```
int ret = ac.DeletePerson(PersonID);
or
```

	`int ret = ac.DeletePerson(m_APE_PersonObj);`

7.8 GetPersonGroupList()

Description:
Use this method to get the person group list.

Parameters:

C#	`APE_Person object list`	This method fills the list of available person groups.

Definition:

C#	`int GetPersonGroupList(List<APE_PersonGroup^>^ m_AcPerGrouplist);`

Example:

C#	`List<APE_PersonGroup>`
	`AcPersonGroupList = new List<APE_PersonGroup>();`
	`int ret = ac.GetPersonGroupList(AcPersonGroupList);`

7.9 GetAuthorizationList()

Description:
Use this method to get the available authorizations [Entrance] list.

Parameters:

C#	`APE_Authorization object list`	This method fills the list of available authorizations.

Definition:

C#	`int GetAuthorizationList(List<APE_Authorization^>` `^m_AcPersonAuthorizationList);`

Example:

C#	`List<APE_Authorization>`
	`AcPersonAuthList new List<APE_ Authorization >();;`
	`int ret = ac.GetAuthorizationList(AcPersonAuthList);`

7.10 GetAuthGroupList()

Description:
Use this method to get the available authorizations [Entrance] group list..

Parameters:

C#	APE_AuthGroup object list	This method fills the list of available authorization groups.

Definition:

C#	int GetAuthGroupList(List<APE_AuthGroup^>^m_AcPerAuthGroupList);

Example:

C#	List<APE_AuthGroup> AcPerAuthGroupList = new List<APE_ AuthGroup >();; int ret = ac.GetAuthGroupList(AcPerAuthGroupList);

7.11 GetEntranceList()

Description:
Use this method to get the available entrance list. It can be used while defining event filter with entrances.

Parameters:

C#	APE_Authorization object list	This method fills the list of available authorizations.

Definition:

C#	int GetEntrancesList(List<APE_Entrance^>^ m_AcEntrancelist);

Example:

C#	List<APE_Entrance> availableEntrancelist = new List<APE_Entrance>(); int val = ac.GetEntrancesList(availableEntrancelist);

7.12 GetEntranceStatesList()

Description:
Use this method to get a list of the door and operation state from all available entrance list. It can be used for initializing the states of all entrances.

Parameters:

C#	APE_EntranceState object list	This method fills the list of available entrances with door and operation state.

Definition:

C#	`int GetEntranceStatesList(List<APE_EntranceState ^>^ m_AcEntranceStatelist);`

Example:

C#	`List<APE_EntranceState> availableEntranceStateList = new List<APE_EntranceState>(); int val = ac.GetEntranceStatesList(availableEntranceStateList);`

7.13 OpenEntranceOnce()

Description:
Use this method to open a specific Entrance.
(Select the Entrance from the available Entrance List.)

Parameters:

C#	APE_Entrance object. **This method will open the entrance.**

Definition:

C#	`int OpenEntranceOnce (APE_Entrance openEntrance);`

Example:

C#	`List<APE_Entrance> availableEntrancelist = new List<APE_Entrance>(); int ret = ac.GetEntrancesList (availableEntranceList); if (ret == 1 && availableEntranceList.Count > 0) { int iSerachIndex = 0 ; // use 0 for example APE_Entrance openEntrance = availableEntranceList[iSerachIndex]; ret = ac.OpenEntranceOnce (openEntrance); }`

7.14 OpenEntranceOnceHandicap()

Description:
Use this method to open a specific Entrance for handicapped persons.
(Select the Entrance from the available Entrance List.)

Parameters:

C#	APE_Entrance object. **This method will open the entrance.**

Definition:

C#	`int OpenEntranceOnceHandicap (APE_Entrance openEntrance);`

Example:

```
C#  List<APE_Entrance> availableEntrancelist = new List<APE_Entrance>();
    int ret = ac.GetEntrancesList (avialableEntranceList);
    if ( ret == 1 && availableEntranceList.Count > 0 )
    {
        int iSerachIndex = 0 ; // use 0 for example
        APE_Entrance openEntrance
=        avialableEntranceList[iSerachIndex];
        ret = ac.OpenEntranceOnceHandicap (openEntrance);
    }
```

7.15 OpenEntrancePermanent()

Description:
Use this method to Open a specific Entrance permanently.
(Select the Entrance from the available Entrance List)

Parameters:

C#	APE_Entrance object. **This method will open the entrance permanently**

Definition:

C#	`i int OpenEntrancePermanent (APE_Entrance openEntrance);`

Example:

```
C#  List<APE_Entrance> availableEntrancelist = new List<APE_Entrance>();
    int ret = ac.GetEntrancesList (availableEntranceList);
    if ( ret == 1 && availableEntranceList.Count > 0 )
    {
        int iSerachIndex = 0 ; // use 0 for example
        APE_Entrance openEntrance
=        availableEntranceList[iSerachIndex];
        ret = ac.OpenEntrancePermanent (openEntrance);
    }
```

7.16 CloseEntrancePermanent()

Description:
Use this method to Close a specific Entrance.
(Select the Entrance from the available Entrance List.)

Parameters:

C#	APE_Entrance object. **This method will close the entrance.**

Definition:

C#	`int OpenEntrancePermanent (APE_Entrance openEntrance);`

Example:

C#	`List<APE_Entrance> availableEntrancelist = new List<APE_Entrance>();`
	`int ret = ac.GetEntrancesList (availableEntranceList);`
	`if ( ret == 1 && availableEntranceList.Count > 0 )`
	`{`
	`  int iSerachIndex = 0 ; // use 0 for example`
	`  APE_Entrance openEntrance`
	`=  availableEntranceList[iSerachIndex];`

7.17 LockEntrance()

Description:
Use this method to Lock a specific Entrance.
(Select the Entrance from the available Entrance List.)

Parameters:

C#	APE_Entrance object. **This method will lock the entrance.**

Definition:

C#	`int LockEntrance (APE_Entrance openEntrance);`

Example:

C#	`List<APE_Entrance> availableEntrancelist = new List<APE_Entrance>();`
	`int ret = ac.GetEntrancesList (availableEntranceList);`
	`if ( ret == 1 && availableEntranceList.Count > 0 )`
	`{`
	`  int iSerachIndex = 0 ; // use 0 for example`
	`  APE_Entrance openEntrance`
	`=  availableEntranceList[iSerachIndex];`
	`  ret = ac.LockEntrance (openEntrance);`
	`}`

7.18 UnlockEntrance()

Description:
Use this method to Unlock a specific Entrance.
(Select the Entrance from the available Entrance List.)

Parameters:

C#	APE_Entrance object. **This method will unlock the entrance.**

Definition:

C#	`int UnlockEntrance (APE_Entrance openEntrance);`

Example:

C#

```
List<APE_Entrance> availableEntrancelist = new List<APE_Entrance>();
int ret = ac.GetEntrancesList (availableEntranceList);
if ( ret == 1 && availableEntranceList.Count > 0 )
{
    int iSerachIndex = 0 ; // use 0 for example
    APE_Entrance openEntrance
=    availableEntranceList[iSerachIndex];
    ret = ac.UnlockEntrance (openEntrance);
}
```

7.19 GetMessageList()

Description:
Use this method to get the available message [event] list. It can be used while defining event filter with message.

Parameters:

C#	`APE_Message object list`	This method fills the list of available messages [event].

Definition:

C#	`int GetMessagesList(List<APE_Message^>^ m_AcMessagelist);`

Example:

C#

```
List<APE_Message> messagelist = new List<APE_Message>();
int val = ac.GetMessagesList(messagelist);
```

7.20 GetNewCardNumber()

Description:
Use this method to get a new card number which does not conflict with other numbers.

Parameters:

C#	`PersonGroup`	Person group ID needed to generate available card number.

Definition:

C#	`int GetNewCardNumber(int personGroup);`

Example:

C#	`int cardNumber = ac.GetNewCardNumber(1);`

7.21 GetPersonCount()

Description:
Use this method to get the count of an available person.

Parameters:
None

Definition:

C#	`int GetPersonCount();`

Example:

C#	`int count = ac.GetPersonCount();`

7.22 ExecuteEventFilter()

Description:
Use this method to apply filter condition to the runtime events as well as to obtain the history.

Notice!
In order to avoid duplication, please clear the local filter store as the new list will be obtained which contains filtered events. For example:
After applying filter like, today's events only and also event generated by reader1: will return a new list of events occurred today on reader1 and also any real-time event which satisfies this filter. And on removing the filter, the SDK provides the list of current date events with no filter and followed by the real-time events.

Parameters:

C#	APE_EventFilter object	This parameter hold the object with the filter definition which will be applied. Apply default values to this object if the user wants to remove the filter.

Definition:

C#	int ExecuteEventFilter(APE_EventFilter^ Filter);

Example:

C#	APE_EventFilter filter = new APE_EventFilter(); filter.FromDate = ”2014/12/28”; filter.ToDate = ”2014/12/28”; filter.LastName = “TestUser”; filter.Company = “Bosch”; filter.CardNo = “456”; filter.Entrances = “Door1;Door2”; filter.Messages = “1;2;3;4”; ac.ExecuteEventFilter(filter);

Default:
Filter is applied for current date only.

7.23 ReceiveEvent()

Description:
Use this method to register external application to receive the events.

Parameters:
None

Definition:

C#	delegate void GetEventHandler(Object^ sender,APEEventArgs^ e); event GetEventHandler^ RecieveEvent;

Example:

C#	ac.RecieveEvent += new APE_SDK.GetEventHandler(DisplayEvent);
	private void DisplayEvent(Object sender,APEEventArgs e)
	{
	APE_Event m_newEvent = e.evt;
	}

7.24 ReceiveEntranceState()

Description:
Use this method to register external application to receive entrance state changes like open/ close door.

Parameters:
None

Definition:

C#
```
delegate void GetEntranceStateHandler (Object^ sender,
APEEntranceStateArgs^ e);
event GetEntranceStateHandler^ RecieveEntranceState;
```

Example:

C#
```
ac.RecieveEntranceState += new
APE_SDK.GetEntranceStateHandler(DisplayEntranceState);
private void DisplayEntranceState (Object sender,
APEEntranceStateArgs e)
{
APE_EntranceState m_newEntranceState = e.est;
}
```

7.25 ReceiveConfigurationChange()

Description:
Use this method to register external application to receive any configuration changes. Instead of RefreshData you can update your person authorization list, authorization group list, entrance list or entrance state list depends on this event.

Parameters:
None

Definition:

C#
```
delegate void GetConfigurationChangeHandler (Object^ sender,
EventArgs^ e);
event GetConfigurationChangeHandler^ RecieveConfigurationChange;
```

Example:

C#
```
ac.RecieveConfigurationChange += new
APE_SDK.GetConfigurationChangeHandler(HandleConfigurationChange)
;
private void HandleConfigurationChange (Object sender,
EventArgs e)
{
```

```
// refresh authorozation List etc.
List<APE_Authorization> AcPersonAuthList new List<APE_
Authorization >;
int ret = ac.GetAuthorizationList(AcPersonAuthList);
```

Notice!
It is recommended to use the trigger for configuration changes. Whenever a change is signaled the used lists (person group list, authorization list, authorization group list, entrance list) can be updated. In this case all your lists are always up-to-date.

7.26 ReceivePersonChange()

Description:
Use this method to register external application to receive any changes of persons data. Instead of RefreshData you can update your persons list on this event.

Parameters:
None

Definition:

C#	delegate void GetPersonChangeHandler (Object^ sender, ApePersonChangeArgs^ e); event GetPersonChangeHandler^ RecievePersonChange;

Example:

C#	ac.RecievePersonChange += new
	APE_SDK.GetPersonChangeHandler(HandlePersonChange);
	private void HandlePersonChange (Object sender, ApePersonChangeArgs e)
	{
	// refresh persons List etc.
	List<APE_PersonsAuthorization> AcPersonAuthList new List<APE_Authorization >;
	int ret = ac.GetAuthorizationList(AcPersonAuthList);

7.27 GetBoschCode1(), GetBoschCode2(), GetBoschCode3()

Description
Use this method to get internal codeNo 1, 2 or 3 in a format like member CodeNo in APE_Event class.

Parameters

C#	APE_Person object	This class object holds the person data from which the code numbers are determined.

Definition

C#	String^ APE_SDK::GetBoschCode1(APE_Person^ apePerson);

Example

C#	APE_Person apePerson = ...;	String Code1 = GetBoschCode1 (apePerson); String Code2 = GetBoschCode2 (apePerson);

		String Code3 = GetBoschCode3 (apePerson);

7.28 GetVersion()

Description:
Use this method to get the current APE_SDK internal version information.

Parameters:

C#	major minor build revision	pointer to an int variable that receives major version number pointer to an int variable that receives minor version number pointer to an int variable that receives build number pointer to an int variable that receives revision number

Example:

C#	int major = 0;
	int minor = 0
	int build = 0;
	int revision = 0;
	ac.GetVersion(ref major, ref minor, ref build, ref revision);

7.29 GetExpectedVersion()

Description:
Use this method to get the expected APE compatibility version information.

Example:

C#	`string sExpVer = ac.GetExpectedVersion();`

7.30 GetServerVersion()

Description:
Use this method to get the the real APE compatibility version information.

Example:

C#	`string sSrvVer = ac.GetServerVersion();`

7.31 RefreshData()

Description:
Use this method to refresh all data (person group list, authorization list, authorization group list, entrance list, message list, etc.) used by the SDK, from APE server.

Parameters:
None

Definition:

C#	int RefreshData();

Example:

C#	int rc = ac.RefreshData();

Notice!
It is recommended to use the trigger for configuration changes. Whenever a change is signaled the used lists (person group list, authorization list, authorization group list, entrance list) can be updated. In this case all your lists are always up-to-date.

8 List of Return Values

SDK_ReturnValue is an enum member variable which contains all the return error codes. The list of return error codes are described in the table below.

Error Code	Description
0	Operation failed
1	Operation succeeded
2	Unable to delete the current user
3	Not authorized to perform the operation
4	Failed to read the configuration file
5	Failed to read the custom data file
6	Error occurred during the Database initialization
7	Error while opening the Database
8	Failed to open a table
9	License tampered
10	Unable to get the person user rights
11	Error while obtaining configuration data
12	Error during reading areas
13	Error during reading all person
14	Error accessing system number table
15	Maximum person count exceeded
16	Error on getting person ID
17	Customer code invalid
18	Invalid country code
19	Invalid card code
20	Invalid card version
21	The specified card code conflicts with his/her existing card code
22	The specified card code conflicts with other person card code
23	The specified card number conflicts with other person card number
24	The specified card number conflicts with his/her existing card number
25	Failed to obtain a new card number
26	Selected person group is invalid
27	Unable to read the Log file
28	SDK license is not activated
29	Cannot delete administrator user

Error Code	Description
30	Demo license expired
31	Codeno/Cardno combination error
32	Login rejected. Too many clients
33	Login rejected. Conflicting compatibility versions
34	General error reading license
35	License details error

9 Appendices

Bosch Security Systems B.V.
Torenallee 49
5617 BA Eindhoven
Netherlands
www.boschsecurity.com