AT Remote Interface Description

en | AT Remote Interface Description

BOSCH

Table of Contents

1 Introduction

1.1 Purpose

The purpose of this document is to describe the remote interface for attendance registration between the CCU and third party software.

1.2 Scope

This document describes the remote interface for attendance registration. It is meant for developers who want to use this remote interface to control the attendance registration application, present in the CCU, remotely. All described functions will be supported in future releases. For a complete description of the System Setup can be referred to [SRS_INF].

1.3 Definitions, Acronyms and Abbreviations

CCU	Central Control Unit. This can be either a single-CCU system or a Multi-CCU system.
DCN	Digital Congress Network
AT	Attendance Registration
ACS	Access Control Services
UnitId	Unit identification, also called unit-number. A unique identification of a unit within the CCU system.
PC	Personal Computer
remote controller	Device (e.g. PC) connected to the CCU which remotely controls a part of the applications present in the CCU.
Present Key	The leftmost softkey of the delegate or chairman unit (softkey 1) with 5 softkeys present, in case the settings and activation for attendance registration request for that functionality

1.4 References

[SRS_INF]	General Remote Interface Description
[SRS_SCSIINF]	SC & SI Remote Interface Description
[USERDOC_AT]	User Manual LBB 3578

This document should be referenced as [SRS_ATINF].

1.5 Overview

Chapter 2 describes the way attendance registration functions inside the CCU. This chapter explains the different parts of attendance registration and the influence of the events upon the state of a unit.
Chapter 3 and chapter 4 describe respectively, the remote functions and the update notifications which can be used to control the attendance of the delegates.
Appendix Appendix A gives an overview of the constants used in combination with the remote functions described in this document.
Appendix Appendix B gives an overview of the possible error's that could be returned upon a remote function request.

2 Internal functioning of Attendance registration

2.1 Introduction

The Attendance Registration application is divided into three closely related parts:

a. Attendance registration
b. Access control
c. Delegate identification

In the following section an explanation is given about the three parts.
Note that if the attendance registration application is not active in the CCU, however the authority settings as present in the delegate database are used to accept or reject actions of the delegates.
For example, when a delegate has no Voting authority, he will not be requested to cast his vote.
The authority settings are part of the delegate database, which should be downloaded using the remote functions as described in [SRS_SCSIINF].

2.1.1 Attendance registration

Attendance registration is an application that allows the remote controller to keep track of the delegates present in the system. To accomplish this the delegate must register himself present by using one of the selectable options:

- Pressing his 'Present key' on his unit. (No leave option)
- Entering his PIN Code on his unit. (No leave option)
- Inserting his ID-card in his or any unit. To leave he withdraws the ID-card.
- Inserting his ID-card in the entrance-unit of the system. To leave he has to insert his ID-card in the exit-unit of the system.

One of the above options is selectable for registration of a delegate.

Note 1: The use of the ID-card can, as an extra option, be combined with entering a pin-code.
Note 2: The ID-card insertion in a unit can be selected to be in his own unit only (fixed seating) or in any unit of the system (free seating). In the latter situation the attendance application connects the current seat to the delegate. The new seat-delegate combination is used within the system.

The attendance registration application keeps track of the delegate which enters the system (become present) and leaves the system (become absent). The differences (if any) are reported to the remote controller every second.

2.1.2 Access Control

Access Control keeps track of the delegate's accessibility for the applications Microphone Management, Intercom and Voting as stored in the delegate database. Note that the contents of the delegate database is controlled by the remote functions available in the System Configuration application. For more information see [SRS_SCSIINF].
A delegate can get control for an application (if he has access according to the authority settings in the delegate database) using one of the following options:

- Entering his PIN Code on his unit.
- Inserting his ID-card in his or any unit.
- Inserting his ID-card in the entrance-unit to get access for his seat as stored in the delegate database. Inserting the ID-card in the exit-unit disables the accessibility.

One of the above options is selectable for access control.

Note: The use of the ID-card can, as an extra option, be combined with entering a pin-code.

The Access Control options are set in combination with the attendance registration options.

2.1.3 Delegate Identification

This functionality provides information about what delegate is seating on which unit. Delegate Identification, i.e. location information, is available as a result of inserting ID Cards in and/or withdrawing them from units. For this functionality neither the attendance registration nor the access control process need to be active. The only restriction is that a namesfile should be downloaded.

When the location of a delegate is determined, the new location is sent to the remote controller. The Delegate Identification functionality keeps track of the location where a delegate is located. The differences (if any) are reported to the remote controller every second.
For Delegate Identification two definitions can be made:

- Located delegate a delegate which resides on a unit.
- Dislocated delegate a delegate which doesn't reside on a unit yet.

A delegate who is assigned a seat in the current names file is using that seat, unless that delegate inserts his card in another unit. In those cases the delegate is a located delegate. If another delegate inserts his card in that particular unit, the delegate which resides default on that unit will become a dislocated delegate.
When a delegate withdraws his card, the delegate will be assigned to his default unit if nobody else is using that unit and the delegate who has withdrawn his card has no pending request to speak, else he will become a dislocated delegate.
The delegate who is by default assigned to the unit from which the card was withdrawn will be assigned to that unit again if the delegate itself is a dislocated delegate. If the delegate is a located delegate, nobody will be assigned to that unit.

2.1.4 Combination Attendance and Access

From the previous sections it will be clear that the settings for attendance registration and access control are combined, because the ways to register and to get access are the same for both parts of the application.
Due to the combination of the settings of the two parts there are some restrictions:

- When the 'Present key' is selected to gain attendance, Access Control cannot be activated.
- When delegates may sit on any chair (Free seating), attendance registration using the 'Present key' is not possible. Also registration and/or Access Control using the PIN Code is not possible with this setting.

2.2 Functioning with parameters

When starting with the attendance application we must use parameters to set the different options. According to the settings made, several events can occur with the DCN-system which influences the presence and access of a delegate.
In this section we define the parameters and create a matrix that defines the changes when a certain event within the system occurs.

2.2.1 State definitions

The state definitions define the current state of a delegate in the DCN system. There will be a state definition per combination of the different settings. The following states are defined:

State item	Value set
Presence	Present or Absent
Location	Located or Anywhere
Authorisation	Functioning or Blocked

Notes

1. Presence is a delegate status identifying if a delegate is present or not.
   Location is a delegate status which reflects on which unit the delegate resides.
   Authorisation is a status identifying if a unit may be used or not by the delegate that currently resides on this unit.
2. When a delegate is marked 'Functioning', the application authorisation stored in the delegate database controls whether access is allowed.

2.2.2 Events definitions

The event definitions shown in the table below are all the events that can influence the presence, authorisation or location of a delegate.

Event	Explanation
Initial / Unit connected	Initial state after activation of settings or state after unit connection
Unit disconnected	Unit disconnects
Present key	Present key pressed on presence menu

PIN Code	PIN Code is successfully entered using the soft-keys This can either be: • PIN Code entered after ID Card insertion (ID Card plus PIN code control) • PIN Code entered directly (PIN Code control)
Insert card in seat	Insert Card in delegate/chairman unit, check if card is inserted in the correct unit and if no other card with the same card code is already present in another unit, check pin code if necessary
Remove card from seat	Remove card from seat after successful "Insert card in seat"
Insert card in Entrance	Insert Card in Entrance unit, check if card in no other unit, check PIN Code if necessary
Insert card in Exit	Insert Card in Exit unit, check if card in no other unit, check PIN Code if necessary

2.2.3 Parameter definitions

Besides the ability to turn on and off the two parts of the attendance application the following parameters are available for setting the options.

Parameter	Explanation
SeatAttend	Determine where the registration must take place. On the seat-unit or on the entrance/exit units.
SeatAccess	Determine if access is allowed on just one seat (as stored in the names file) or on any seat. Seat access 'None' means that no names file is currently opened.
ControlType	Determine how the delegate must register himself to the system. Possible options are: Present Key, PIN Code, ID Card and ID Card plus PIN Code.

2.2.4 Event / state matrix

The table on the next page presents the event / state matrix for the different settings of the parameters.

Attendance	Access	Seat Attend	Seat Access	Control-Type	Initial / Unit connected	Unit Disconnected	Present Key	PIN Code	Insert Card in Seat	Remove Card from Seat	Insert Card in Entrance	Insert Card in Exit
OFF	OFF	-	-	-	Absent Anywhere Functioning	Absent Anywhere Blocked			Absent Located Functioning	Absent Anywhere Functioning		
	ON	ENTRANCE EXIT[1]	ONE_SEAT	IDCARD (_PINCODE)	Absent Located Blocked	Absent Located Blocked			Absent Located Functioning	Absent Anywhere Blocked		
			ANY_SEAT	IDCARD (_PINCODE)	Absent Anywhere Blocked	Absent Anywhere Blocked			Absent Located Functioning	Absent Anywhere Blocked		
		SEAT	ONE_SEAT	PIN CODE	Absent Located Blocked	Absent Located Blocked		Absent Located Functioning	Absent Located (No change)	Absent Anywhere (No change)		
				IDCARD (_PINCODE)	Absent Located Blocked	Absent Located Blocked			Absent Located Functioning	Absent Anywhere Blocked		
			ANY_SEAT	IDCARD (_PINCODE)	Absent Anywhere Blocked	Absent Anywhere Blocked			Absent Located Functioning	Absent Anywhere Blocked		
ON	OFF	ENTRANCE EXIT	ONE_SEAT	IDCARD (_PINCODE)	Absent Located Functioning	(No change) Located Blocked			(No change) Located Functioning	(No change) Anywhere Functioning	Present Located Functioning	Absent Located Functioning
			ANY_SEAT	IDCARD (_PINCODE)	Absent[2] Anywhere Functioning	(No change) Anywhere Blocked			(No change) Located Functioning	(No change) Anywhere Functioning	Present Anywhere Functioning	Absent Anywhere Functioning
		SEAT	NONE	PRESENT-KEY	Absent Anywhere Functioning	Absent Anywhere Blocked	Present Anywhere Functioning					
			ONE_SEAT	PRESENT-KEY	Absent Located Functioning	Absent Located Blocked	Present Located Functioning		(No change) Located Functioning	(No change) Anywhere Functioning		

Attendance	Access	Seat Attend	Seat Access	Control-Type	Initial / Unit connected	Unit Disconnected	Present Key	PIN Code	Insert Card in Seat	Remove Card from Seat	Insert Card in Entrance	Insert Card in Exit
ON	OFF	SEAT	ONE_SEAT	PIN CODE	Absent Located Functioning	Absent Located Blocked		Present Located Functioning	(No change) Located Functioning	(No change) Anywhere Functioning		
				IDCARD (_PINCODE)	Absent Located Functioning	Absent Located Blocked			Present Located Functioning	Absent Anywhere Functioning		
			ANY_SEAT	IDCARD (_PINCODE)	Absent Anywhere Functioning	Absent Anywhere Blocked			Present Located Functioning	Absent Anywhere Functioning		
	ON	ENTRANCE EXIT	ONE_SEAT	IDCARD (_PINCODE)	Absent[2] Located Blocked	(No change) Located Blocked			(No change) Located Functioning	(No change) Anywhere Functioning	Present Located Functioning	Absent Located Blocked
			ANY_SEAT	IDCARD (_PINCODE)	Absent[2] Anywhere Blocked	(No change) Anywhere Blocked			(No change)[3] Located[3] Functioning[3]	(No change) Anywhere Blocked	Present Anywhere Blocked	Absent Anywhere Blocked
		SEAT	ONE_SEAT	PIN CODE	Absent Located Blocked	Absent Located Blocked		Present Located Functioning	(No change) Located Functioning	(No change) Anywhere Functioning		
				IDCARD (_PINCODE)	Absent Located Blocked	Absent Located Blocked			Present Located Functioning	Absent Located Blocked		
			ANY_SEAT	IDCARD (_PINCODE)	Absent Anywhere Blocked	Absent Anywhere Blocked			Present Located Functioning	Absent Anywhere Blocked		

The notes mentioned in the table are:

1. There are several rows showing the same states on the same events (e.g., Attendance Off, Access On and Seat Attend on Entrance-Exit units is functional the same for both Seat Access on One-seat and Seat Access on Any-seat). Although it seems doubled information, all allowed combinations are shown, amongst others to understand the changes in settings..
2. Initial State, No change at connection of the unit.
3. The delegate must be present to come to this state, otherwise no acceptation.

Combinations of settings that are not present in the table are not allowed.

In case that no delegate database is downloaded into the CCU settings for ID-card or PIN Code are not possible. There is simply no information about which delegate has which ID-card or PIN Code.

Therefor, when no delegate database is downloaded into the CCU, only one event / state combination is legal:

Attendance	Access	Seat Attend	Seat Access	Control-Type	Initial / Unit connected	Unit Disconnected	Present Key	PIN Code	Insert Card in Seat	Remove Card from Seat	Insert Card in Entrance	Insert Card in Exit
ON	OFF	SEAT	NONE	PRESENT KEY	Absent Anywhere Functioning	Absent Anywhere Blocked	Present Anywhere Functioning					

Note that in this situation the activation of the present-key only registers the seat, because the system does not know which delegate should be seated on that seat. Thus, in this specific situation no delegate/unit information will be sent to the remote controller. Only the total number of present reports is sent.

3 Remote Functions

3.1 Introduction

This chapter describes the various remote functions needed to control the attendance registration application inside the CCU. A global description of the remote function handling is described in [SRS_INF].
In the [SRS_SCSIINF] is stated that the CCU can operate in multiple modes. The use of the AT remote function is restricted to the “Congress Mode”. For more information about the various modes see [SRS_SCSIINF].

3.1.1 Remote function item explanation

Each description consists of the following items:

- **Purpose**
  A global description of the purpose of the function.
- **Parameter structure for the function**
  The input parameters needed to fulfil the function. When the function requires no parameters, no structure is described here.
- **Response structure from the function**
  The output information coming from the function called. This information is only valid when the ‘wError’ field of the received response information equals AT_E_NOERROR.
- **Error codes returned**
  The error values returned in the ‘wError’ field of the response information. All possible error codes are described in appendix Appendix B.
- **Update notifications**
  The update notifications that are generated during the execution of the remote function. When there are no notifications generated, then this part will be omitted.
- **Related functions**
  The related function in conjunction with the function described. It refers to other remote functions and to related update notifications.

3.2 Attendance/Access functions

3.2.1 AT_C_START_AT_APP

Purpose

Indicate the CCU that the remote controller wants to communicate with the AT application inside the CCU. Depending on the control-type passed the remote controller gets the opportunity to start attendance registration and/or access control. When no control is needed, but the remote controller likes to know which delegates are present (i.e. for microphone display information), the remote controller can monitor the presence changes from the CCU.
When you omit the execution of this remote function, all other remote functions have no effect and will return an error.

Parameter structure for the function

The function requires the following structure as parameters.

```
typedef struct
{
    BYTEbyRemoteControlType;
} AT_T_APPCONTROL;
```

where:

byRemoteControlType Identify what function the remote controller likes to perform in combination with the attendance application. Valid values are:

- AT_C_APP_CONTROL The remote controller likes to have full control over the attendance registration application. This full control implies the right to change the attendance registration settings.
- AT_C_APP_MONITOR The remote controller only wants to monitor the presence changes. No

control of the settings is allowed.

Note that the second start of the application (without a stop) always results in an error. This implies that you cannot change from 'control' to 'monitor' mode by calling the AT_C_START_AT_APP again. You have to call the function AT_C_STOP_AT_APP first to stop the previous mode.

Response structure from the function

The function has no response parameters.

Error codes returned

AT_E_NOERROR
AT_E_INCONTROL_OTHER_CHANNEL
AT_E_INCONTROL_THIS_CHANNEL
AT_E_INMONITOR_THIS_CHANNEL
AT_E_ILLEGAL_CONTROL_TYPE

Related functions

AT_C_STOP_AT_APP

3.2.2 AT_C_STOP_AT_APP

Purpose

Indicate the CCU that the remote controller no longer requires to communicate with the AT application inside the CCU. When the remote controller which has the control ability stops the communication, the CCU takes over the control for AT and turns attendance registration and access control off if they were still on.

Note that: Upon communication lost this function will be activated, if AT_C_START_AT_APP was activated.

Parameter structure for the function

The function has no additional parameters.

Response structure from the function

The function has no response parameters.

Error codes returned

AT_E_NOERROR
AT_E_APP_NOT_STARTED

Related functions

AT_C_START_AT_APP

3.2.3 AT_C_STORE_SETTING

Purpose

This function allows the remote controller to pass the new setting for attendance registration and access control to the attendance registration application on the CCU. The attendance registration application checks the validity of the parameters passed and stores the new settings.
Note that this function may only be called if both attendance registration and access control are off. See the AT_C_ACTIVATE function (§3.2.4).

Parameter structure for the function

The function requires the following structure as parameter:

```
typedef struct
{
    BYTE    bySeatAttend;
    BYTE    bySeatAccess;
    BYTE    byControlType;
} AT_T_SETTINGS;
```

where:

bySeatAttend — Identify on which type of unit attendance registration will take place. The setting is one of the following:

- AT_C_SEAT
- AT_C_ENTRANCE_EXIT

bySeatAccess — Identify if a delegate can only use his own assigned unit or also another unit. The setting is one of the following:

- AT_C_ANY_SEAT
- AT_C_ONE_SEAT

byControlType — Identify how attendance registration and/or access control will take place. The setting is one of the following:

- AT_C_PRESENTKEY
- AT_C_PINCODE
- AT_C_IDCARD
- AT_C_IDCARD_PINCODE

The meaning of the different parameter setting is described in §2.2.3.

Response structure from the function

The function has no response parameters.

Error codes returned

AT_E_NOERROR
AT_E_APP_NOT_STARTED
AT_E_STORE_SETTING_FAILED
AT_E_CHANGE_NOT_ALLOWED
AT_E_NOT_INCONTROL

Related functions

AT_C_ACTIVATE
AT_C_HANDLE_IDENTIFICATION

3.2.4 AT_C_ACTIVATE

purpose

This function allows the remote controller to start/stop attendance registration and/or access control. As long as attendance registration and/or access control is on, the CCU will send update notifications of type AT_C_SEND_TOTAL_REGISTRATION to the remote controller. Update notifications are sent upon state changes due to actions from the delegates on the units.

Parameter structure for the function

The function requires the following structure as parameter:

```
typedef struct
{
    BOOLEAN     bAttendanceOn;
    BOOLEAN     bAccessOn;
} AT_T_ACTIVATE;
```

where:

bAttendanceOn	Indication if attendance registration must be on or off
bAccessOn	Indication if access control must be on or off

Response structure from the function

The function has no response parameters.

Error codes returned

AT_E_NOERROR
AT_E_APP_NOT_STARTED
AT_E_NOT_INCONTROL
AT_E_ACTIVATION_NOT_ALLOWED

Update notifications

AT_C_SEND_INDIV_REGISTRATION
AT_C_SEND_TOTAL_REGISTRATION

Related Functions

AT_C_STORE_SETTING
AT_C_HANDLE_IDENTIFICATION

3.2.5 AT_C_HANDLE_IDENTIFICATION

Purpose

This function allows the remote controller to do the registration with his own equipment. After the local registration on the remote controller, he should pass the registered delegate to the DCN-system.
The registration from the remote controller emulates the insertion of the ID-card in the entrance- or exit- unit. Therefore the ID-card code and (optional) the PIN-code must be passed along with this function.
Note that both the ID-card-codes and the PIN-codes are downloaded from the remote controller into the CCU during the download of the delegate database (see [SRS_SCSIINF] for details).
Together with the registration of the delegates, at the unit, on which the delegate resides, all LED's will be turned on if the delegate becomes present. The LED's are turned off again when the delegate is registered absent.

Parameter structure for the function

The function requires the following structure as parameter:

```
typedef struct
{
    WORD    wEvent;
    WORD    wFillLevel;
    AT_T_DEL_IDENTIFICATION    tDelIdenfication [AT_C_MAX_REGISTRATION];
} AT_T_IDENTIFICATION_REC;
```

where the AT_T_DEL_IDENTIFICATION is defined as:

```
typdef struct
{
    DWORD       dwCardCode;
    WORD    wPinCode;
} AT_T_DEL_IDENTIFICATION;
```

where:

wEvent — Identify on which type of unit attendance registration will take place. The setting is one of the following:

- ACSC_EVENT_INSERT_CARD_ENTRANCE
- ACSC_EVENT_INSERT_CARD_EXIT

wFillLevel — Number of delegates in *tDelIdentification* (ranges from 1 to AT_C_MAX_REGISTRATION). If more than AT_C_MAX_REGISTRATION delegates should be registered this function must be called more than once.

tDelIdentification [] — Structure containing the delegate identification.

dwCardCode — ID-Card code of the delegate that should be registered. Valid ID-card codes are in the range 1-MAX_CARD_CODE (the ID-card code must be unique for every delegate in the DCN-system).

wPinCode — PIN-code of the delegate hat should be registered. The PIN-code is only used when the 'Control-Type' is set to the value AT_C_IDCARD_PINCODE (see §3.2.3)

Valid PIN-codes are in the range 111-55555, whereby each digit must be in the range of 1-5. Set the field wPinCode to 0 (zero) if PIN-codes are not used.
The number of digits to be used is also stored into the delegate database. (PIN-codes do not have to be unique.)

This function will handle the request only if the function AT_C_STORE_SETTINGS is called before with the settings:

bySeatAttend	AT_C_ENTRANCE_EXIT
bySeatAccess	AT_C_ONE_SEAT
byControlType	AT_C_IDCARD or AT_C_IDCARD_PINCODE

and the function AT_C_ACTIVATE is called before to activate either Attendance Registration or Access Control or both.

Response structure from the function

The function has no response parameters.

Error codes returned

AT_E_NOERROR
AT_E_HANDLE_IDENTIFICATION_FAILED
AT_E_APP_NOT_STARTED
AT_E_SETTING_NOT_CORRECT
AT_E_NOT_INCONTROL
AT_E_ILLEGAL_EVENT
AT_E_ILLEGAL_ARRAY_SIZE

Update notifications

AT_C_SEND_INDIV_REGISTRATION
AT_C_SEND_TOTAL_REGISTRATION

Related functions

AT_C_STORE_SETTING
AT_C_ACTIVATE

3.2.6 AT_C_GET_INDIV_REGISTRATION

Purpose

This function allows the remote controller to retrieve the current registration status of each individual delegate. The function is meant for remote controllers who called the function AT_C_START_AT_APP with AT_C_APP_MONITOR as control type while attendance registration and/or access control was already activated.
The function enables the remote controller to create his own start-up status of the delegate registrations which is to be used to handle the registration changes send by the application specific update notifications.

Parameter structure for the function

The function requires the following structure as parameter:

```
typedef struct
{
    WORD    wClusterIndex;
} AT_T_GET_REGISTRATION;
```

where:

wClusterIndex	An index that indicates which cluster of delegate registration information is to be retrieved. When wClusterIndex is 0 (zero), the response structure contains the first cluster, with a maximum of AT_C_MAX_DELEGATE, of delegate registration information. When wClusterIndex is 1 (one), the second cluster is returned etc.

Response structure from the function

The function returns the following structure:

```
typedef struct
{
    WORD    wFillLevel;
    AT_T_DEL_ATTEND  tDelegate[AT_C_MAX_DELEGATE];
} AT_T_REGISTER_INDIV;
```

where the AT_T_DEL_ATTEND is defined as:

```
typedef struct
{
    WORD    wUnitId;
    WORD    wDelegateId;
    BYTE    byAttend;
} AT_T_DEL_ATTEND;
```

where:

wFillLevel		Number of delegates in *tDelegate* (maximum of AT_C_MAX_DELEGATE) If wFillLevel is less than AT_C_MAX_DELEGATE, then the last cluster with delegate registration information is returned.
tDelegate		Structure containing the delegate information.
	wUnitId	Unit on which the delegate is located. The *wUnitId* can be the value DBSC_EMPTY_UNIT when the delegate is not located anywhere.
	wDelegateId	Delegate for which the presence status is given.
	byAttend	Presence status of the delegate. The setting is one of the following: • AT_C_NOSTATUS • AT_C_ATTEND • AT_C_LEAVE • AT_C_ATTEND_NOCHANGE

- AT_C_LEAVE_NOCHANGE

How the presence status is determined can be seen in the following status diagram:

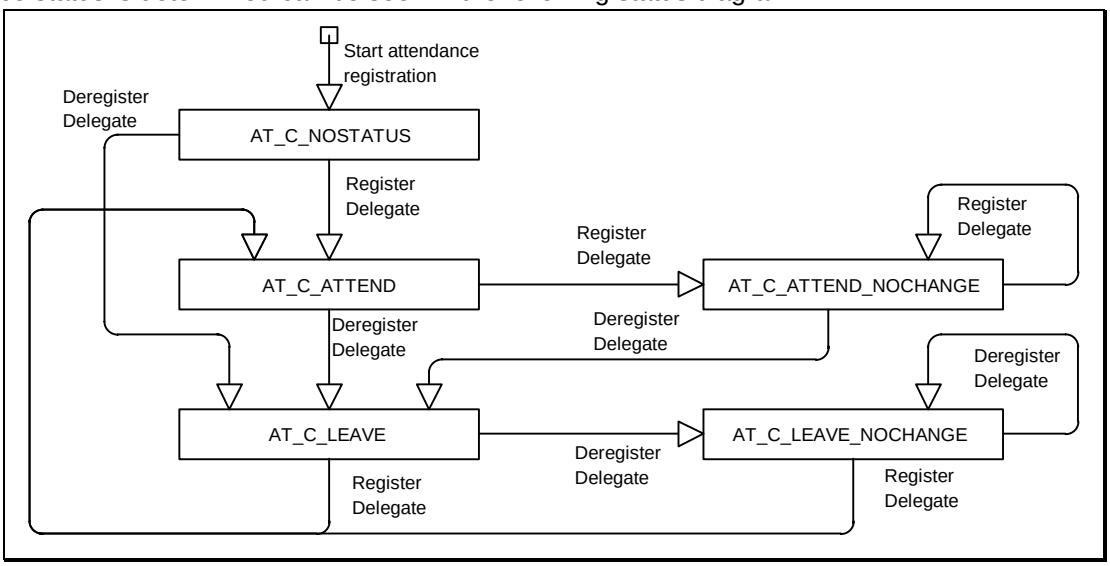

Figure 1 Presence status changes diagram

Internally it is possible that a delegate which is already present, will be registered present again. In this case he inserts his ID-card in an another unit, which implies that the delegate changed seat. This seat change is also reported to the remote controller using this update notification. His status will then be changed to AT_C_ATTEND_NOCHANGE to inform that the 'presence' has not changed. The same situation can occur when the delegate has already left the system.

Error codes returned

AT_E_NOERROR
AT_E_APP_NOT_STARTED

Update notifications

AT_C_SEND_INDIV_REGISTRATION
AT_C_SEND_TOTAL_REGISTRATION

Related functions

AT_C_START_AT_APP

4 Update Notifications

4.1 Introduction

This chapter describes the various update notifications send by the CCU. All the update notifications of the AT application are listed in this chapter. A global description of notifications is made in [SRS_INF].

4.1.1 Preconditions

The update notifications coming from the AT application use the UnitId and DelegateId to connect each other. The valid unitId's in the DCN-system can be queried and the DelegateId's can be set using remote functions described in [SRS_SCSIINF].

4.1.2 Notification item explanation

Each description consists of the following items:

- **Purpose**
  A global description of the purpose of the notification.
- **Notify structure with this update**
  The information passed with the update notification.
- **Related functions**
  The related function in conjunction with the notification described.

4.2 Attendance Registration and Access Control notifications

4.2.1 AT_C_SEND_INDIV_REGISTRATION

Purpose

Notify the remote controller the individual status of the delegates which (de)registers themselves. The presence and location results will be sent every second if changes have been detected on the CCU. Also the initial state (directly after activation) is sent to the remote controller using this notification. The notification is sent to every controller who started the attendance registration application with AT_C_START_AT_APP before.

Notify structure with this update

The update comes with the same structure as used for the response of the remote function AT_C_GET_INDIV_REGISTRATION (section 3.2.6).
Note that only the changes are sent to the remote controller.

Related functions

AT_C_STORE_SETTING
AT_C_ACTIVATE
AT_C_HANDLE_IDENTIFICATION

4.2.2 AT_C_SEND_TOTAL_REGISTRATION

Purpose

Notify the remote controller the total number of present and absent delegates. This information will be sent every second by the CCU if changes have been detected on the CCU. Also the initial totals (directly after activation) is sent to the remote controller using this notification.
These results will only be sent if attendance registration is activated.

Notify structure with this update

The update comes with the following structure:

```
typedef struct
{
    WORD    wAttend;
    WORD    wLeave;
} AT_T_REGISTER_TOTAL;
```

where:

wAttend	Number of delegates who have registered themselves present.
wLeave	Number of delegates who are known in the delegate database and who are not registered yet.

Related functions

AT_C_STORE_SETTING
AT_C_ACTIVATE
AT_C_HANDLE_IDENTIFICATION

APPENDIX A. VALUES OF THE DEFINES

In this document a lot of definitions are given, which have values connected to them. In this appendix all defines will be connected to their values;
The values are presented in 'C'-syntax

```
#define AT_C_START_AT_APP(0x0901)
#define AT_C_STOP_AT_APP (0x0902)
#define AT_C_STORE_SETTING              (0x0903)
#define AT_C_ACTIVATE    (0x0904)
#define AT_C_HANDLE_IDENTIFICATION     (0x0905)
#define AT_C_GET_INDIV_REGISTRATION    (0x0906)

#define AT_C_SEND_INDIV_REGISTRATION   (0x090A)
#define AT_C_SEND_TOTAL_REGISTRATION   (0x090B)

#define AT_C_APP_CONTROL 1
#define AT_C_APP_MONITOR 2

#define AT_C_SEAT    1
#define AT_C_ENTRANCE_EXIT             2

#define AT_C_ANY_SEAT    1
#define AT_C_ONE_SEAT    2

#define AT_C_PRESENTKEY  1
#define AT_C_IDCARD  2
#define AT_C_IDCARD_PINCODE            3
#define AT_C_PINCODE 4

#define AT_C_NOSTATUS    0
#define AT_C_ATTEND  1
#define AT_C_LEAVE   2
#define AT_C_ATTEND_NOCHANGE           3
#define AT_C_LEAVE_NOCHANGE            4

#define AT_C_MAX_DELEGATE250
#define AT_C_MAX_REGISTRATION          50

#define ACSC_EVENT_INSERT_CARD_ENTRANCE      5
#define ACSC_EVENT_INSERT_CARD_EXIT    6

#define DBSC_EMPTY_UNIT  0xFFFF
#define DBSC_EMPTY_DELEGATE            0xFFFF

#define TRUE 1
#define FALSE    0

#define MAX_CARD_CODE    999999999L
```

APPENDIX B. ERROR CODES

Responses returned upon a remote function request contain an error field ('wError'). In this appendix an overview is given of the possible errors and their values.

Attendance Registration Error code Explanation	Value
AT_E_NOERROR The execution of the remote function was successful.	0
AT_E_APP_NOT_STARTED The remote controller has not called the AT_C_START_AT_APP yet. Therefore any remote function call to the attendance registration application fails with this error.	2305
AT_E_STORE_SETTING_FAILED Settings or a combination of settings is not correct.	2306
AT_E_HANDLE_IDENTIFICATION_FAILED The eventId, the ID-card code and/or length of PIN-code are not correct to handle the requested action.	2314
AT_E_SETTING_NOT_CORRECT Settings are not correct to handle the requested action.	2315
AT_E_INCONTROL_OTHER_CHANNEL The AT_C_START_AT_APP function could not finish successfully because the attendance application is already controlled by another remote controller using another channel.	2316
AT_E_INCONTROL_THIS_CHANNEL The attendance application is already under control by this remote controller (on the same channel). Probably you have called the AT_C_START_AT_APP function twice.	2317
AT_E_INMONITOR_THIS_CHANNEL The attendance application is already monitored by this remote controller (on the same channel). Probably you have called the AT_C_START_AT_APP function twice.	2318
AT_E_NOT_INCONTROL The remote function is not allowed, because this remote controller has no control over the attendance registration application.	2319
AT_E_CHANGE_NOT_ALLOWED A change of setting (even if they are the same as the previous call) is not allowed, because attendance registration and/or access control is currently active.	2321
AT_E_ACTIVATION_NOT_ALLOWED The settings made by the remote function AT_C_STORE_SETTING are conflict with the activation or deactivation of attendance registration and/or access control. See chapter 2 for more information.	2322
AT_E_ILLEGAL_CONTROL_TYPE The control-type passed to the function AT_C_START_AT_APP is not within range of valid values (see appendix Appendix A for the correct control-type values).	2333
AT_E_ILLEGAL_EVENT The event-type passed to the function AT_C_HANDLE_IDENTIFICATION is not within range of valid values (see appendix Appendix A for the correct event values).	2334
AT_E_ILLEGAL_ARRAY_SIZE The fill-level passed along with the function AT_C_HANDLE_IDENTIFICATION exceeds the maximum array size.	2335

APPENDIX C. EXAMPLES

In the examples below the remote functions and update notifications, that are defined in this document as constant values for the wFnId parameter of the message (see [SRS_INF]), are presented as functions described in a 'C' syntax. The parameter structures of these functions are according the input, output or notify structures described in the appropriate section.
For every function is assumed that the function will create the required input parameter structure, transport the parameters to the CCU and waits for the result information coming from the CCU.
For both the remote functions as the update notifications the same names are used as their identifier, but without the constant mark "C", some "_" and using mixed case names.
For example remote function AT_C_STORE_SETTING shall be referenced as function:

AT_StoreSetting (&tSettings);

C.1. Using Attendance Registration and Access Control

This example shows how the remote controller can perform attendance registration with the entrance- and exit units by using ID Cards.
For this example we have defined the following DCN-system:

- A conference hall equipped with delegate units without ID-card readers
- Entrance and Exit units are present.
- The seat-assignment has been done by the remote controller.
- A delegate database is downloaded into the CCU.

Using this system we like to use the ID-cards for registration and access control for all delegates. Because the system does not have an ID-card reader in the units, we use card-readers in the entrance- and exit units to register the delegates.
First the remote controller must register himself to the AT application.

```
error = AT_StartATApp (AT_C_APP_CONTROL);
switch (error)
{
    case AT_E_INCONTROL_THIS_CHANNEL:
    /* I have the attendance registration app already under control */
    /* Is that correct? Is the remote controller restarted? */
    /* For the moment assume to be correct and continue */
    .........
    break;

    case AT_E_INCONTROL_OTHER_CHANNEL:
    /* Another remote controller has control over the attendance registration app */
    /* report error and terminate */
    .........
    return;

    case AT_E_INMONITOR_THIS_CHANNEL:
    /* I tried to open the application for control, but it seems that I have the */
    /* attendance registration application already opened for Monitoring attendance.
*/
    /* report error and terminate */
    .........
    return;

    case AT_E_NOERROR:
    /* function ended succesful, continue */
    break;

    default:
    /* some unexpected error occurred. */
    /* report the error */
    .........
    break;
}
```

We now have control over the attendance registration application and may change the settings, but first the input parameter structure must be filled in:

```
AT_T_SETTINGS     tSettings;

tSettings.bySeatAttend      = AT_C_ENTRANCE_EXIT;
tSettings.bySeatAccess      = AT_C_ONE_SEAT;
```

```
tSettings.byControlType            = AT_C_IDCARD;


error = AT_StoreSetting (&tSettings);
if (error != AT_E_NOERROR)
{
 /* do error handling */
}
```

Starting attendance registration and access control will be done by calling the following function:

```
AT_T_ACTIVATE     tActivate;

tActivate.bAttendanceOn     = TRUE;
tActivate.bAccessOn   = TRUE;

error = AT_Activate (&tActivate);
if (error != AT_E_NOERROR)
{
 /* do error handling */
}
```

The CCU is now running attendance registration and access control. When a delegate inserts his ID-card into an entrance unit, the AT application on the CCU sends an “individual registration” and “total registration” notification.
This result in the following two functions:

```
void AT_SendIndivRegistration (AT_T_REGISTER_INDIV *tIndivResults)
{
 WORD wIndex;

 /* get presence of delegates */
 for (wIndex = 0; wIndex < tIndivResults->wFillLevel; wIndex++)
 {
     /* handle the presency of each delegate separately */
 }
}

void AT_SendTotalRegistration (AT_T_REGISTER_TOTAL *tTotalResults)
{
 /* update the local results with the new total present and absent information
    from the CCU */
}
```

When the remote controller is also equipped with a card-reader, then the delegates may use that card-reader to register themselves. In that specific case the remote controller reads the ID-card and registers the delegate to the Attendance application by using the AT_C_HANDLE_IDENTIFICATION remote function.
For example when two delegates with card-code 16824 and 6823 have registered themselves using the remote controller, the remote controller performs the following actions:

```
AT_T_IDENTIFICATION_REC  tIdentification;

tIdentification.wEvent   = ACSC_EVENT_INSERT_CARD_ENTRANCE;
tIdentification.wFillLevel     = 2;
tIdentification.tDelIdentification [0].dwCardCode = 16824;
tIdentification.tDelIdentification [0].wPinCode   = 0;      /* not used */
tIdentification.tDelIdentification [1].dwCardCode = 6823;
tIdentification.tDelIdentification [1].wPinCode   = 0;      /* not used */

wError = AT_HandleIdentification (&tIdentification);
if (wError != AT_E_NOERROR)
{
    /* do error handling */
}
```

Finally, when the congress is ended, we can stop the Attendance registration and Access control by calling:

```
AT_T_ACTIVATE     tActivate;

tActivate.bAttendanceOn     = FALSE;
tActivate.bAccessOn   = FALSE;

error = AT_Activate (&tActivate);
if (error != AT_E_NOERROR)
{
```

```
 /* do error handling */
}
```

Now the control can be given back to the CCU by calling the following function:

```
error = AT_StopATApp ();
if (error != AT_E_NOERROR)
{
 /* do error handling */
}
```

For more information please visit www.boschsecuritysystems.com

December 2003 | AT Remote Interface Description

BOSCH