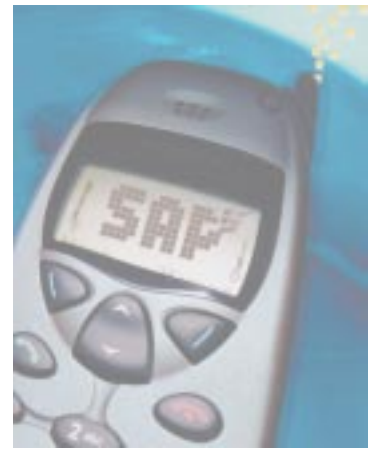
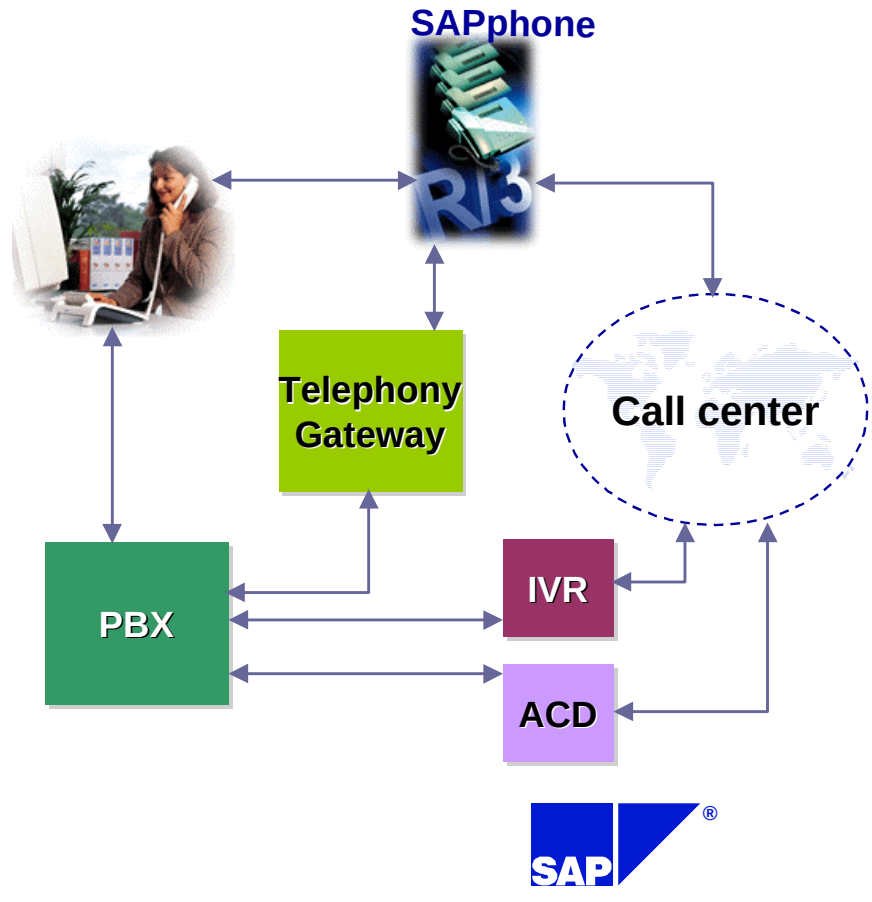


SAPphone -
Telephone integration in R/3

Susanne Gottlieb
SAP AG

Telephone Integration in R/3

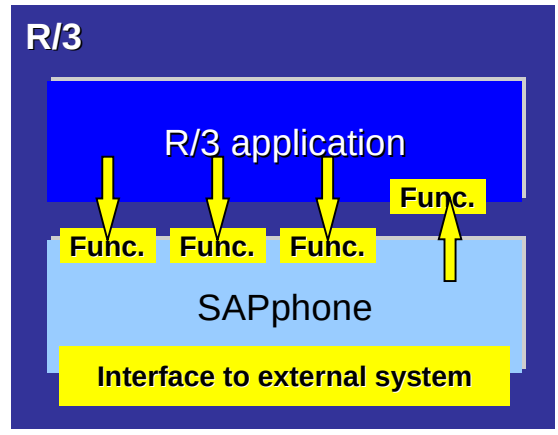
- Linking R/3 directly to a PBX or CTI system
- Integration with R/3 applications



SAPphone - A Basis Service

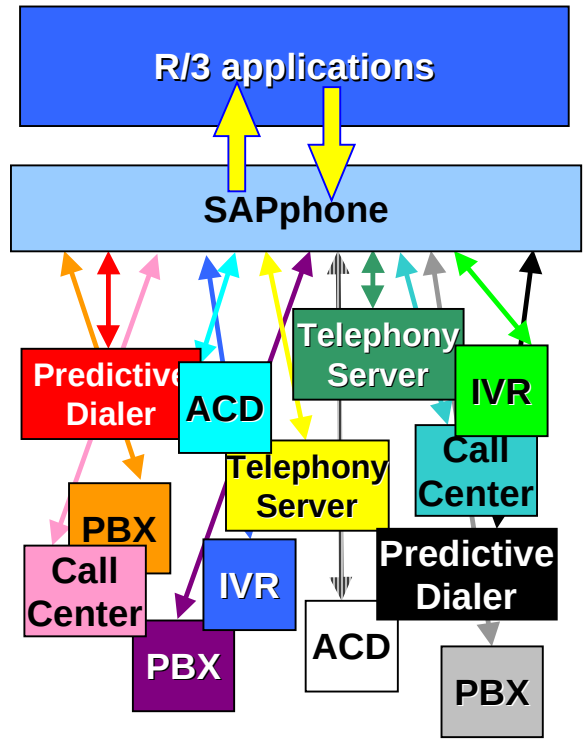
SAPphone is ...

- **An integral part of R/3**
- **R/3's interface to CTI systems**
- **A service for R/3 applications**
  - **applications call SAPphone functions to execute call control functions**
  - **applications are triggered by SAPphone for incoming calls**



SAPphone - Goals

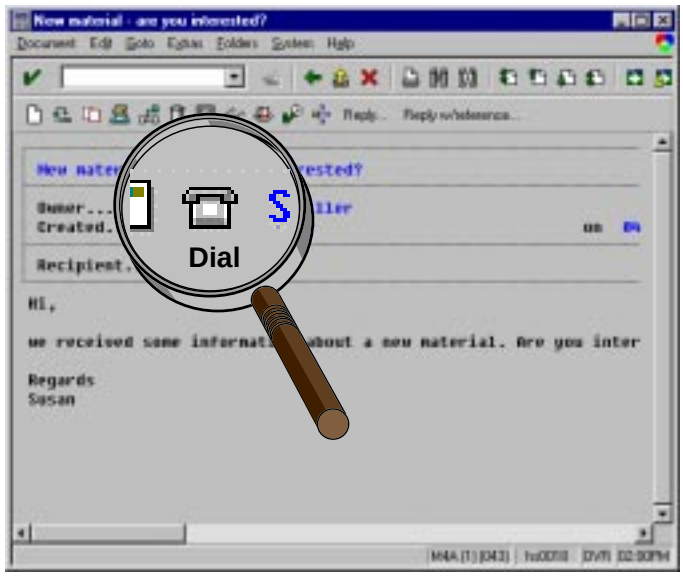
- **Integration in R/3 business processes**
  - **Telephone functions**
  - **Call-related data**
  - **Status information**
- **Independence from external CTI systems**
  - **Multiple CTI systems**



RFC Interface

- Set of function modules for interaction between R/3 and external CTI system
- Published interface
  - Description available
- General availability
  - PBX vendors
  - Call center
  - Third-party
- Interface certification will be available soon

Support for Outgoing Calls

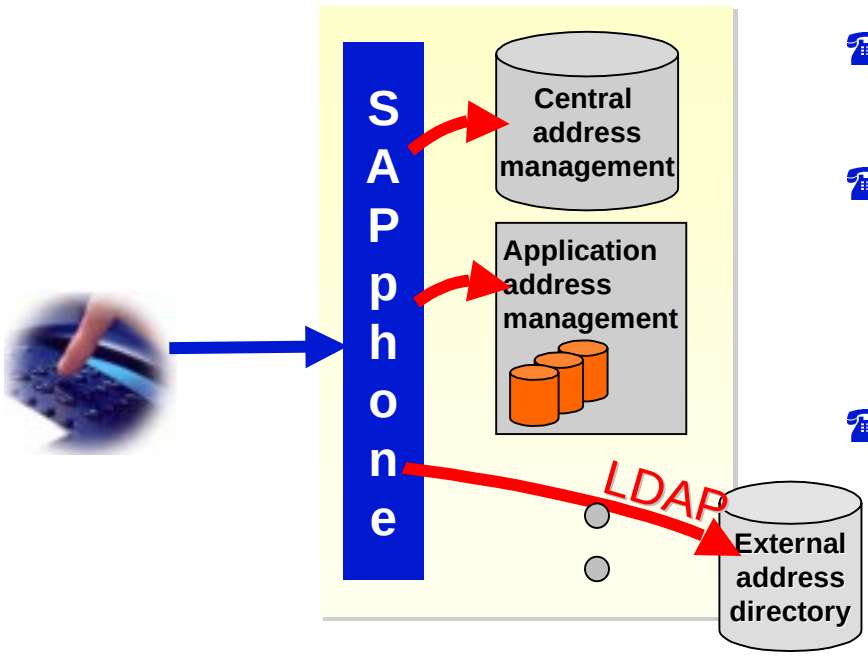


- Use of R/3 address data
- Automatic number formatting
- List of alternative phone numbers
- Redial
- Workflow and SAPoffice integration

Implementation: Step-by-Step

1. **Screen:**
   - Add DIAL button on screen
2. **PBO:**
   - Check if work center is configured for SAPphone: if not, hide DIAL button
3. **Program:**
   - Read telephone number or address number of call receipient
4. **PAI:**
   - Add case for DIAL in ok-code module
   - Call function OUTGOING_CALL_PROCESS

Caller Identification

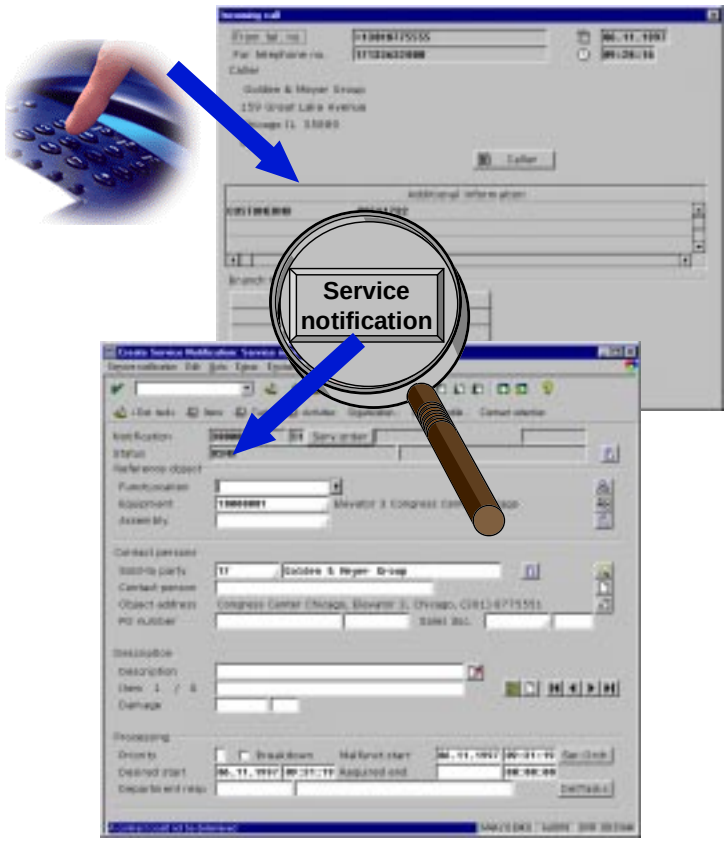


- R/3 central address management
- Support for LDAP-compatible address directories is planned
- Multiple address directories can be connected

3 Ways to Process Incoming Calls

- Wait for incoming call
  - As used in Customer Interaction Center
  - Only suitable for call centers
  - Incoming call is displayed immediately on screen (no user action)
- Button
  - To get call data within the current application
  - To be implemented within the application (button, call SAPphone function module and data integration)
- **Popup**
  - **Caller data and selection of startable applications**

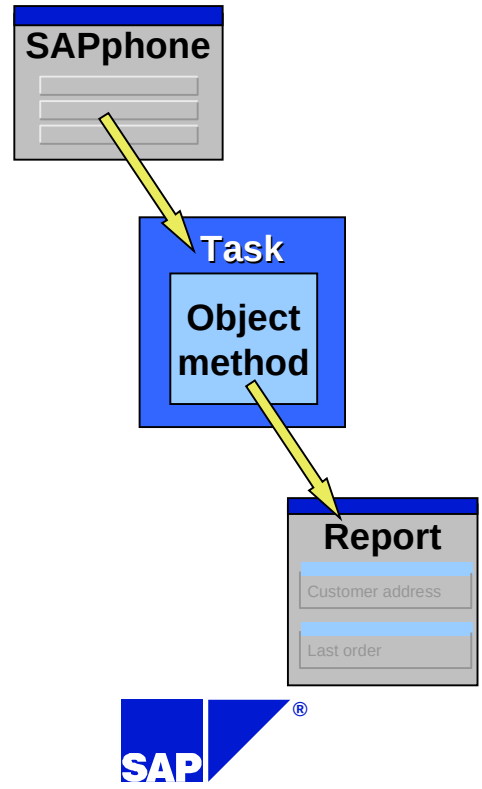
Display Incoming Call



- **Smart caller identification**
- **Display of caller address**
- **Start application**
- **Workflow integration**

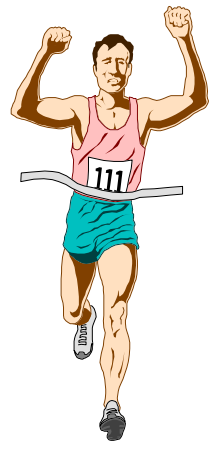
Add Application to Task List

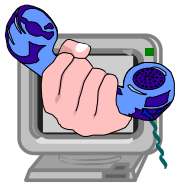
- Use existing business object method or create new method encapsulating the application
- Create task (single-step or workflow) using the business object method
- Mark task as ‘suitable for incoming calls’
  - In the task definition under SAPphone tab page
- Maintain agent assignment



Questions?

Thank you for your attention





SAPphone
Telephone Integration in R/3

Integrating functions and data related to incoming and outgoing calls in R/3 applications

Version 1.0

Copyright

History

Version	Date
1.0	**August 13, 1998**

Contents

1 Outgoing Calls

1.1 Functionality

The function module described in the following section allows outgoing calls to be initiated from within existing transactions or lists. The function provides a dialog box which offers additional functions and allows the user to enter or change the phone number to be dialed.

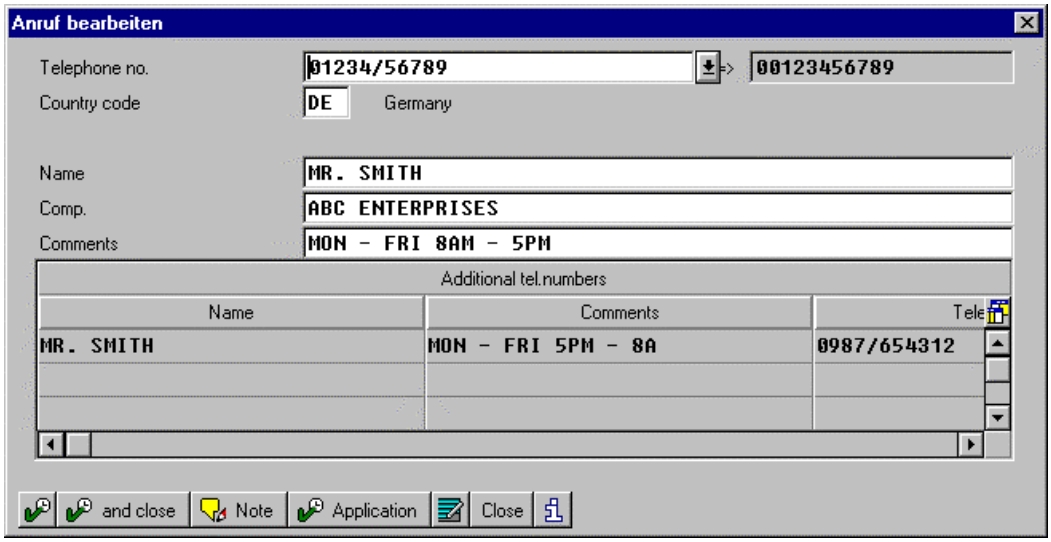


Figure 1-1: Dialog box for initiating outgoing calls

- The phone number to be dialed and the corresponding country code are always displayed. They are not mandatory import parameters for the function module and can also only be entered manually from the dialog box. However, the values are required to execute the following functions: *Dial*, *Note*, *Application* and the optional function. The dialog box can also be used to enter and modify the displayed number. This change does not affect the master data record from which the number was originally retrieved and only modifies the number to be dialed in this case. The user instantly sees the number which results from the conversion to canonical format or to the number to be dialed and can correct the number before the call is initiated, if necessary.

- The fields *Name*, *Company* and *Remark* are always displayed. They can be imported from the calling application or entered within the dialog box. The values are used for user information and are transferred when the application callback function is called.. They are also exported back to the calling application when the dialog box is closed.

- The caller of the function can transfer a list of any length containing name-value pairs which are displayed in the dialog box. These values are only for information purposes and cannot be changed. If the number of lines exceeds the size of the dialog box, scrolling is possible.

From 4.0

- The caller of the function can transfer a list of any length containing alternative phone numbers which are displayed in the table in the dialog box. By double-clicking on an entry, the user can transfer data to the other fields in the dialog box and can use this data to initiate a call or execute any of the other available functions.

- When the *Refresh* button is pressed, the contents of the dialog box are updated, which means that the number is converted again if changes have been made to the phone number or country code and the country name is determined again.

- The *Dial* (execute) button initiates a call to the displayed number. The dialog box remains on screen to allow the user to terminate the call using the *Hang up* button and to allow the user to execute certain functions (*Note*, *Applications* and any application-specific callback functions), even after the call has been completed. The *Dial and close* button initiates a call to the displayed number and closes the dialog box.

- The *Hang up* button refers to the call and only terminates the connection. The dialog box remains on screen. The button is only displayed if a call is both active and recognized by the system, i.e. the call was initiated from the dialog box or a call ID was transferred when the function was called.

- If the *Note* function is called, a SAPoffice document which references the call is created by attaching an object of type VOICE_CALL to the document. By double-clicking on the object or executing the object, the function described above is called and the dialog box is displayed, so that a call can be initiated. The telephone number is entered automatically. The *Note* function is available as standard, but can be hidden (see the interface for the function module OUTGOING_CALL_PROCESS).

From 4.0

- The *Applications* function is used to display a list of application tasks (single-step or multistep tasks, e.g. workflows), from which the user can select and start a task. To be included in the list, the task has to meet two requirements:
  1. The user must have authorization to execute the task (the authorizations are maintained in task maintenance via agent assignment).
  2. The task must be marked as *Suitable for SAPphone outgoing calls*. This attribute can be set in task maintenance.

  Note: If both requirements have been met and the task is still not in the list, you should refresh the organizational environment, e.g. the task overview in SAPphone system administration.

  The following data is passed to the task as container elements:

Name	Type	Description
NUMBER	SPH_TELNO-TELNO	Phone number of the person to call
COUNTRY	SPH_TELNO-COUNTRY	Country code of the person to call
NAME	SPH_TELNO-NAME	Name of the person to call
COMPANY	SPH_TELNO-COMPANY	Company of the person to call
REMARK	SPH_TELNO-REMARK	Remark about the person to call

Up to 3.1

- The *Work item* function creates a work item for the call and sends the work item to the user's integrated inbox, so that the call can be initiated at a later time by executing the work item. The button is available as standard, but can be hidden if necessary.

- The text of the optional button (*Optional*), as well as the functionality, are defined by the caller of the function. If no function is specified, the dialog box is closed immediately when the button is pressed and the export parameter 'OPT_BUTTON_PRESSED' is set to 'X'. If a function is defined, this function is executed when the button is pressed. In this case, the dialog box remains on screen. (→ interface for callback function).

  Possibilities for using the optional button:
  - ⇒ With callback function: when the button is pressed, a list of alternative call partners is displayed. When a call partner is selected, the relevant data (phone number, country code, etc.) is transferred to the dialog box.
  - ⇒ With callback function: to support the processing of a call list, the optional button can be used to determine the next call partner and transfer the relevant data to the dialog box. If the functionality is implemented via a callback function module, the dialog box can remain on screen the whole time and does not have to be re-opened each time.
  - ⇒ With callback function: when the button is pressed, more detailed information about the current call partner can be provided by the application.

- The *Close* function closes the dialog box. This does not affect any current dialing operations or existing connections.

- All buttons can be hidden via an exclude table (see the interface for the function module OUTGOING_CALL_PROCESS).

1.2 Settings / Administration

Before users can take advantage of the functionality described in the previous section, certain administrative tasks have to be performed. For further information, you should consult the SAPphone online documentation, which can be found under *BC-Basis* → *Basis Services* → *Communications interface*.

1.3 Integrating the Functionality in an Application

1.3.1 Procedure for integrating the *Dial* function in a transaction

1. Screen:
Dial button on screen.
Please read the ergonomics recommendations for more information about texts and icons.

2. PBO:
Insert a module for calling the function module TELEPHONY_ACTIVE_GET. If the export parameter ACTIVE does not have the value 'X', the *Dial* button should be hidden.
Purpose: the function should be hidden if the current work center does not offer telephony support.

3. Program:
Maintain the variables for telephone number and country code (or the address number from central address management, from Rel. 4.0 onwards). Other fields can also be maintained, e.g. optional button, callback function, alternative telephone numbers, depending on how the functions are to be configured.

4. PAI:
Insert a new case for dialing in the module for evaluating the function code.
Call the function module OUTGOING_CALL_PROCESS from there.

1.3.2 Interfaces

The interfaces of the function modules for initiating calls and checking the telephony-active flag, as well as the optional callback function module provided by the application, are described in the following table:

1.3.2.1 OUTGOING_CALL_PROCESS

From 4.0

Import	Type	Description
TEL_NUMBER	SPH_CALL-NO_DIALED	Number to call *optional*
COUNTRY_TO	T005-LAND1	Country code of the call recipient *optional*
ADDRESS_NUMBER	ADDR_KEY	Address number from central address management *optional*
BUTTON_TEXT	SPHFIELDS-NAME	Text for optional application button *optional*
DISPLAY_POPUP	SPH_PCTONO-IN_ACTIVE	Flag specifying whether dialog box is displayed *optional; Default*: dialog box is displayed
POPUP_TITLE	SPHFIELDS-NAME	Title of the dialog box *optional, Default:* 'Initiate call'
TEST_MODE	SPH_PCTONO-IN_ACTIVE	Flag specifying whether the function is called in test mode *optional; Default:* no test mode
CALLBACK_FUNCTION	RS38L-NAME	Function which is called when the optional button is pressed *optional; Default:* not called
ACTION	SPH_PCTONO-OUT_ACTIVE	Only evaluated if DISPLAY_POPUP is not set: flag specifying whether the call is initiated or terminated *optional; Default:* Dial (D)

NAME_IN	SPHFIELDS-VALUE	Name of call recipient *optional*
COMPANY_IN	SPHFIELDS-VALUE	Company of call recipient *optional*
REMARK_IN	SPHFIELDS-VALUE	Remark about call *optional*
HANDLE _IN	TSKCS-HANDLE	Call ID *optional;* only needed if the call is to be terminated without a dialog box

From 4.0

Export	Type	Description
NUMBER_CALLED	TSKPA-TELNR	Number dialed
OPT_BUTTON_PRESSED	SPH_PCTONO-IN_ACTIVE	Optional button was pressed
NAME_OUT	SPHFIELDS-VALUE	Name of call recipient
COMPANY_OUT	SPHFIELDS-VALUE	Company of call recipient
REMARK_OUT	SPHFIELDS-VALUE	Remark about call
HANDLE_OUT	TSKCS-HANDLE	Call ID
DIAL_BUTTON_PRESSED	SPH_PCTONO-IN_ACTIVE	*Dial* button was pressed, the command to establish a connection was successfully transferred to the external software (this does not guarantee that a connection could be established)

From 4.0

From 4.0

Tables	Type	Description
FIELDS	SPHFIELDS	List of additional information to be displayed on screen *optional*
EXCLUDED	HRSFCODES	Function codes for buttons which are to be hidden *optional*
TELNO_LIST	SPH_TELNO	Alternative telephone numbers which are displayed on screen in a table and can be selected by double-click *optional*
ADDR_LIST	ADDR_KEY	List of alternative call partners, identified by the address number from central address management
EXTCALLS	SPH_CSTATE	List of calls which are currently active for this extension

From 4.0

Exception	Description
ERROR_WS	Error in external software
SYSTEM_FAILURE	System error, triggered by the RFC layer
COMMUNICATION_FAILURE	Communication error, triggered by the RFC layer
NUMBER_INVALID	Invalid number (only raised when function is called without a dialog box. With a dialog box, an error message is displayed directly and the number must be entered again.)
ERROR_TERMINAL_GET	User is logged on to more than one work center and the user's own extension cannot therefore be determined
CONFIGURATION_ERROR	The configuration contains an error
SP_ERROR	A general error occurred, see the error message for further information

Rel 3.1/4.0

Rel 3.1/4.0

Rel 3.1/4.0

Rel 3.1/4.0

Rel 3.1/4.0

Rel 3.1/4.0

From 4.5

From 4.5

SP_CTI_ERROR	An error occurred in the external software, see the error message for further information

1.3.2.2 TELEPHONY_ACTIVE_GET

Export	Type	Description
ACTIVE	CHAR 1	Flag specifying whether the work center supports telephony integration

1.3.2.3 Callback function

Changing	Type	Description
TEL_NUMBER	SPH_CALL-NO_DIALED	Telephone number of call recipient
COUNTRY	T005-LAND1	Country code
NAME	SPHFIELDS-VALUE	Name of call recipient
COMPANY	SPHFIELDS-VALUE	Company of call recipient
REMARK	SPHFIELDS-VALUE	Remark about call

Tables	Type	Description
FIELDS	SPHFIELDS	List of additional information displayed in dialog box
TELNOLIST	SPH_TELNO	Alternative phone numbers displayed in dialog box

From 4.0

1.3.3 Note about ergonomics

For ergonomic reasons, access to the telephony functionality should be the same from all application screens, which allows users to recognize the new icon and associate the correct function with the icon.

When implementing the telephony function in a transaction, there are two different cases to consider:

1. A field with the telephone number is displayed on the screen from which the *Dial* function is to be implemented.
2. There is no telephone number field on screen, but one business partner is specified.

In the first case, the button (telephone icon without text) should be displayed as close as possible to the field containing the telephone number.

Advantages: Can be accessed via the keyboard
Functionality is visible
Conforms to standard solutions for other functions, e.g. long text

Disadvantages: Requires more space

In the second case, the button design should be dependent on the position on screen:

Application toolbar: A telephone icon without additional text is sufficient (saves space and is not language-dependent)

Initial screen: If there is more space available, the button can contain the *Dial* text, along with the telephone icon

2 Incoming Calls

The following text describes how incoming calls are processed in SAPphone, as well as how applications can integrate their own functions to allow specialized processing.

In the call center environment, there is another option for processing incoming calls: the Customer Interaction Center (CIC). The CIC is an integrated user interface for a call center agent which allows the agent, for example, to manage calls (answer, transfer, consult, hang up, etc.) and start applications. The integration of applications into the CIC is not described in this document.

2.1 Functionality

At present, there are three options when processing the data from incoming calls in R/3:

1. An incoming call starts a dialog box, in which the caller data is displayed and from which applications can be started. From R/3 Rel. 4.0, applications can also be started directly. The application to be started is defined by the individual user.
2. When an incoming call arrives, the user presses a button in the application. The data for the current call is passed to the application and can be displayed on screen.
3. The user presses a button and switches to 'wait mode'. The screen is blocked until the next call for the work center is received. The screen is then refreshed automatically without user interaction and displays the call data. This technique is only used for CICs (available from Rel. 4.5A).

From Rel. 4.5A onwards, the term 'incoming calls' can also include telephone calls which are made in the context of a telephone campaign. These calls are actually outbound calls (to the customer), but are presented to the user as inbound calls.

2.1.1 Starting a dialog box

- ***Up to R/3 Rel. 4.0B***
  An incoming call starts the function module INCOMING_CALL on the R/3 application server via RFC. The function module is either called from the telephony server (central architecture) or from the PC of the call recipient (local or client-server architecture). The following data is passed to the function module:
  - Number dialed (e.g. service hotline)
  - Number to which the call was transferred (work center extension)
  - Number of caller (if available)
  - Flag specifying whether an express notification should be send (not used from Rel. 4.0 onwards)
  - Server ID (new from Rel. 4.0)
  - Additional options (new from Rel. 4.0, see interface description)
  - Additional data, e.g. data collected previously by an IVR system and passed to the application (from Rel. 3.1H onwards)

- ***From R/3 Rel. 4.5A onwards***
  An incoming call starts the function module SPS_NEW_CALL on the R/3 application server via RFC. The function module can be started directly from the telephony server (central architecture) or from the PC of the call recipient (local or client-server architecture). The following data is passed to the function module:
  - Number dialed (e.g. service hotline)
  - Own extension
  - Number of call partner (if available)
  - Server ID
  - Call ID
  - Additional options (see interface description)
  - Additional data, e.g. data collected previously by an IVR system
  - A list of calls which are currently active for your extension

- The call recipient is determined from the number dialed and the telephony server ID (from Rel. 4.0A onwards, when available). If the ‘Display call’ or ‘Temporarily store call data’ (for collecting data within an application) options have been selected in the user-specific settings (from Rel. 4.0A: if SAPphone user settings exist for the user), the call is processed as follows, otherwise the processing ends here.

- The address data areas (function modules for finding and reading data from application master data tables) assigned to the call recipient are searched for the caller name, company and address using the caller number as a reference. The telephone number is used in the format in which the number was received from the telephony software (there is no defined format). If the user settings of the call recipient have been maintained accordingly (i.e. the option ‘Temporarily store call data’ has been selected in Rel. 3.1, or user settings simply exist in Rel. 4.0), the call data is stored in table INDX (before Rel. 4.5A) and can be retrieved by an application. From Rel. 4.5A onwards, the call data is stored by the external software. If the user has also (or only) selected the option ‘Display call’ (or ‘Display call’ or ‘Start task’ from Rel.4.0A onwards), the call is processed as follows.

- In the case of central installations, the call recipient receives an express notification about the call when the next interaction with the system occurs. If the caller could be identified, the name, company and telephone number are displayed in this dialog box. By double-clicking on the icon on the express notification, the user can access a detailed call display screen.

Figure 2-1: Express notification about an incoming call

In the case of local or client-server installations, the express notification is not displayed and call processing continues immediately.

- A dialog box is started, displaying the caller address (if the caller could be identified) and a list of applications which can be started from here (see 2.2.1 for more information about the composition of this list). The user can select and start an application. The data determined so far is passed to the application (see interface).
  From Rel. 4.0 onwards:
  The following options, which can be selected in the user-specific settings, are also available:
  - ⇒ If an application is to be started immediately, the application is called, without the SAPphone dialog box being displayed, and the data which could be determined is passed to the application (address number of the caller, date and time of the call, caller and call recipient phone numbers).
  - ⇒ If, in addition, the option ‘Start only if the caller data could be determined’ is selected, the application is only started if the caller could be identified. If no caller data could be determined, the SAPphone dialog box for incoming calls is displayed.

  Users can also enter an address for an incoming call manually via the dialog box if no address could be found or if the current address is incorrect.

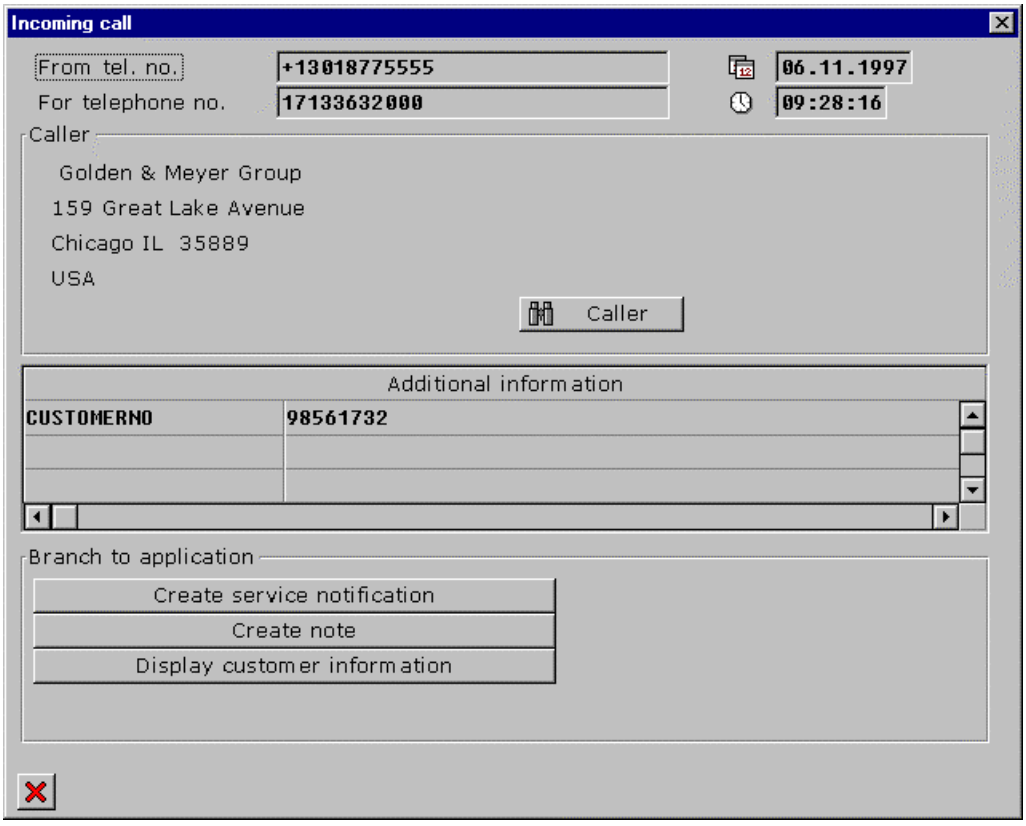


Figure 2-2: Dialog box for incoming calls

2.1.2 Retrieving call data in an application

A button must be implemented on screen in the application. If the user presses the button when a call is received, the following happens:

Up to Rel. 4.0B, the application calls the function module INCOMING_CALL_GET. The function module reads the data for the current call and transfers the data to the application. Call data is stored in table INDX when the call is received by the R/3 System (only one set of call data can be stored for each work center). See also: 2.1.1.

From Rel. 4.5A onwards, call data is no longer stored in the R/3 System, but in the external CTI software. In this case, the application calls the function module SP_CALL_GET, which calls the external software and receives a list of all calls currently available for the work center (more than one call per work center is now possible). The function module displays a list of calls to the user, who can select one call from the list. The data for the call is then transferred to the application.

2.1.3 Wait mode

Wait mode is only suitable for interfaces which are specially designed for call center use (e.g. CIC). When the user activates wait mode by pressing a button, a function of the external CTI software is called. However, this function is called immediately, but only returns when the next incoming call arrives. The call data is then returned immediately. In this case, the data or the user settings are not evaluated. The application can now process the data further and is responsible for evaluating the data. There are some function modules to support this task, e.g. a function module which identifies the caller. These function modules are listed below.

SP_GETCALLSTATETEXT	Determines the description of the numerical status code of the call (e.g. ringing)
SP_ANI_TO_ADDRESS	Allows the assignment of the incoming telephone number to an address from central address management which is selected manually. This is required for automatic caller identification next time.
SP_REMOTE_PARTY_IDENTIFY	Identifies the caller using the telephone number. SAPphone user settings and standard mechanisms are used.

2.2 Integrating Applications

There are several links between applications and the processing of incoming calls:

- Applications can be started by the recipient of an incoming call
- Applications can be started directly by an incoming call (new in Rel. 4.0A)
- Application tables containing address data are searched for caller data
- Applications can 'collect' call data

The following section describes what an application must do to support all of the above scenarios. Of course, applications can also support only some of the scenarios (e.g. caller identification, starting tasks).

2.2.1 Applications to be started when an incoming call is received

2.2.1.1 Implementation in R/3 Rel. 3.1

The list of applications which are available to the user when an incoming call arrives is maintained in the SAPphone user settings and stored in table SOGR. To be included in the list, an application must provide a business object type for which a special method is defined. This method must encapsulate the functionality. The application objects must support the interface IFINCCALL, which specifies the name, interface and type of the method to be executed, so that all applications can be handled in the same way. The method must also meet the following requirements:

- Instance-independent, i.e. the method must not use the object key. The address number of the caller, if available, is transferred to the method via the container. However, the method should also provide a solution if the address number is not available. The method can also be added to existing object types.
- Defined container definition:

Name	Type
TEL_NUMBER	SPH_CALLER-TELNO
COUNTRY	SPH_CALLER-COUNTRY
ADDRESS	SPH_CALLER-ADDRESS
Tables	
ADDITIONAL_DATA	SPH_FIELDS

- The method name must be INC_CALL_PROCESS, the description (short text) should be overwritten and should describe the application as clearly as possible, because this text is displayed wherever the task can be selected.
- Interface IFRECEIVE must be supported to allow the assignment of the task to a user.

The user can select and start an application. Creating a note with reference to the call is always possible.

2.2.1.2 Implementation from R/3 Rel. 4.0 onwards

To provide a function which can be executed by the user when an incoming call is received, the application has to provide tasks (single-step or multistep). The tasks can encapsulate any object types and methods or be complete workflows. Call data is passed to the tasks in the form of container elements (see chapter 2.4.2.4)

If the applications wish to access the call data, the relevant fields must be defined in the task container and the binding with the object method must be defined. However, the application should always be able to run without this data, i.e. support its own search engines, because the data will not always be available.

The task must meet two criteria in order to be displayed for selection:

1. The user must be authorized to execute the task.
2. The task must be marked as *Suitable for SAPphone incoming calls*. This attribute can be set in task maintenance.

Note: If both requirements have been met and the task is still not in the list, you should refresh the organizational environment, e.g. the task overview in SAPphone system administration.

2.2.2 Application tables with address data

To find the data for a caller from the phone number, the application must be informed of the address data tables (see client-wide settings). Every application for which the address tables are to be linked to SAPphone must provide two function modules (see interface description):

- Search function module
  The function module determines the address number of the caller using the phone number
- Read function module
  The function module reads the address data using the address number and returns the data in a defined format

2.2.3 Retrieving an incoming call in an application

Up to Rel. 4.0B, if SAPphone settings exist for a user (in Rel. 3.1: if the user has selected 'Temporarily store call data' in the user settings), the call data for an incoming call is always stored in table INDX, which allows applications to access the data. To access the call, the application calls the function module INCOMING_CALL_GET. Firstly, the work center for the current user is determined. The call data for the most recent incoming call is stored in table INDX under the work center ID. The data which is saved includes the incoming phone number, the address number of the call partner and the table (or address data area) in which the data record was found, if the call data could be determined.

From Rel. 4.5A, the application uses the function module SP_CALL_GET to access the data for a current call. The call data is no longer stored in the R/3 System, but in the external CTI software which is connected to SAPphone. It is now also possible to store data for several calls which are all active at the same time (e.g. one call on hold, another as part of a conference call). When the function is called, a list of all active calls (if more than one active call exists) is displayed to the user, from which one call can be selected and the relevant data is then transferred to the application.

2.3 Settings / Administration

Before a user can use the functionality described above, certain administrative settings must be entered. For more information, you should read the SAPphone online documentation, which can be found under *BC-Basis* → *Basis Services* → *Communications interface*.

2.4 Interfaces

2.4.1 Interfaces to external telephony software

2.4.1.1 INCOMING_CALL (up to R/3 Rel. 4.0)

Import	Type	Description
NUMBER_DIALED	SPH_CALL-NO_DIALED	Number dialed
NUMBER_TRANSFERRED_TO	SPH_CALL-NO_TRANSF	Number to which the call was transferred
NUMBER_CALLING	SPH_CALL-NO_CALLING	Calling number
EXPRESS	SPH_SERVER-LOCALSERV	Send express notification (not used from Rel. 4.0 onwards)
PROTOCOL	SPH_SERVER-LOCALSERV	Internal use only
SERVER_NAME	SPH_SERVID-SERVER_ID	Server ID, used with the number dialed to determine the call recipient.
OPTIONS	SPHOPTIONS	Structure for transferring additional control parameters (set first character in first field to 'X' to identify the caller and return the export parameter without starting call processing).
HANDLE	TSKCS-HANDLE	Call ID

! *From 4.0*

	Export	Type	Description
From 3.1H	NAME	SPHFIELDS_VALUE	Caller name
From 3.1H	COMPANY	SPHFIELDS_VALUE	Caller company
From 3.1H	CITY	SPHFIELDS_VALUE	Caller city
From 3.1H	COUNTRY	SPHFIELDS_VALUE	Caller country

	Tables	Type	Description
From 3.1H	ADDITIONAL_DATA	SPHFIELDS	Structure to transfer additional data as name-value pairs which are sent to the application.

Exception	Description
RECEIVER_NOT_FOUND	Call recipient could not be determined
ERROR_EXPRESS_SEND	An error occurred while sending the express notification
ERROR_NUMBER_DIALED	The number dialed is not valid

2.4.1.2 SPS_NEW_CALL (from R/3 Rel. 4.5)

Import	Type	Description
REMOTE_PARTY	SP_TELNO	Number of call partner
DNIS	SP_TELNO	Number which was originally dialed
OWN_EXTENSION	SP_TELNO	Own number
SERVER_NAME	SP_SERV_ID	Server ID, used with the number dialed to determine the call recipient.
HANDLE	SP_HANDLE	Call ID, assigned by the external software. The call can be further processed (e.g. transferred) via this ID later.
OPTIONS	SPHOPTIONS	Structure for transferring additional control parameters (set first character in first field to ‘X’ to identify the caller and return the export parameter without starting call processing).
PROTOCOL	SP_BOOLEAN	Internal use only

Export	Type	Description
NAME	SPHFIELDS_VALUE	Caller name
COMPANY	SPHFIELDS_VALUE	Caller company
CITY	SPHFIELDS_VALUE	Caller city
COUNTRY	SPHFIELDS_VALUE	Caller country

Tables	Type	Description
EXTCALLS	SPH_CSTATE	List of currently active calls for your extension.

CALLDATA	SPH_IOCONT	Structure to transfer additional data as name-value pairs.

Exception	Description
ERROR_EXPRESS_SEND	An error occurred while sending the express notification
ERROR_NUMBER_DIALED	The number dialed is not valid

2.4.2 Interfaces to the application

2.4.2.1 INCOMING_CALL_GET (up to R/3 Rel. 4.0)

Export	Type	Description
CALLER_ADDRESS	ADDR_KEY	Caller address number (new type in Rel. 4.0)
CALLER_TELNO	SPH_CALL-NO_CALLING	Caller phone number
CALLER_TABLE	SPH_ATIC-ID	Address data area in which the caller was found
CALL_DATE	SY-DATUM	Date of call
CALL_TIME	SY-UZEIT	Time of call
NUMBER_CALLED	SPH_CALL-NO_DIALED	Number of call recipient
CALL_ENDED	SPH_PCTONO-IN_CALL	Indicator that the call has been completed
CALLER_DATA	SPH_CALLER	Caller data of first identified caller

! *From 4.0*

Tables	Type	Description
ADDITIONAL_DATA	SPHFIELDS	Additional data as name-value pairs
CALLER_DATA_TAB	SPH_CALLER	List of all identified addresses for incoming number

'rom 3.1H

From 4.0

Exception	Description
NO_DATA_FOUND	There is no data available

2.4.2.2 SP_CALL_GET (from R/3 Rel. 4.5)

Export	Type	Description
CALLSTATE	SPH_CSTATE	Call status in numeric form
TIME	SPH_CALL-CALL_TIME	Time of call
DATE	SPH_CALL-CALL_DATE	Date of call

Tables	Type	Description
CALLINFO	SPH_CINFO	Information about the call
CALLDATA	SPH_IOCONT	Additional data, e.g. collected by an IVR
REMOTE_PARTY_ADDRESS_LIST	SPH_CALLER	List of caller data

Exception	Description
SP_ERROR	General error occurred, see error message for more information
SP_CTI_FAILURE	Error occurred in external CTI system, see error message for more information
NO_CALL	No calls available at present

2.4.2.3 Interface for application-specific object methods (in R/3 Rel. 3.1)

INC_CALL_PROCESS

Import	Type	Description
ADDRESS	SADR-ADRNR	Address number
TEL_NUMBER	SPH_CALL-NO_CALLING	Caller number
COUNTRY	T005-LAND1	Caller country code

Tables	Type	Description
ADDITIONAL_DATA	SPHFIELDS	Additional data as name-value pairs

From 3.1H

2.4.2.4 Interface of SAPphone enabled task (from Rel. 4.0 on)

Import	Type	Description
CALLING_NUMBER	SPH_CALL-NO_CALLING	Calling number
CALLER_ADDRESS	ADDR_KEY	Address number of first identified caller
CALL_DATE	SPH_CALL-CALL_DATE	Date of call
CALL_TIME	SPH_CALL-CALL_TIME	Time of call
CALLED_NUMBER	SPH_CALL-NO_DIALED	Number dialed
CALLER_ADDRESS_AREA	SPH_CALLER-ATIC_ID	Address data area of first identified caller
CALL_HANDLE	SPH_CALL-HANDLE	Call ID

From 4.5A

From 4.5A

Tables	Type	Description
CALLER_DATA	SPH_CALLER	List of all identified callers
ADDITIONAL_DATA	SPHFIELDS	Additional data as name-value pairs

2.4.2.5 Interface for functions to search for / read caller data

2.4.2.5.1 Function to search for data (in R/3 Rel. 3.1)

Import	Type	Description
TELNUMBER	SPH_CALL-NO_CALLING	Caller number
COUNTRY	T005-LAND1	Caller country code

Export	Type	Description
ADDRESS	SPH_CALL-ADDRESS	Caller address number

Exception	Description
NO_DATA_FOUND	There is no data available

2.4.2.5.2 Function to search for data (from R/3 Rel. 4.0)

Export	Type	Description
TELNUMBER	SPH_CALL-NO_CALLING	Caller number

Tables	Type	Description
ADDRESS	ADDR_KEY	Caller address number

Exception	Description
NO_DATA_FOUND	There is no data available

2.4.2.5.3 Function to read data

Import	Type	Description
ADDRESS	ADDR_KEY	Caller address number (new type from Rel. 4.0 onwards)

Export	Type	Description
CALLER	SPH_CALLER	Caller data (type changed in Rel. 4.0)

Exception	Description
NOT_FOUND	There is no data available

2.4.2.6 Function modules to read data attached to a call

2.4.2.6.1 SP_GETINFODIR

Import	Type	Description
HANDLE	SP_HANDLE	Call ID

Tables	Type	Description
OBJECT_DIR	SPH_IODIR	Directory of all objects for this call (several instances per object are possible)
OBJECT_DIR_UNIQUE	SPH_UIODIR	Directory of all objects for this call (each object listed only once)

Exceptions	Description
SP_ERROR	General error occurred, see error message for more information
SP_CTI_FAILURE	Error occurred in external CTI software, see error message for more information

2.4.2.6.2 SP_GETINFO

Import	Type	Description
HANDLE	SP_HANDLE	Call ID
OBJECT_NAME	SP_IONAM	Name of information object
OBJECT_NUMBER	SP_IONUM	Instance of information object

Tables	Type	Description
IOBJECT	SPH_IOBJ	List of all key-value pairs for the object

Exceptions	Description
SP_ERROR	General error occurred, see error message for more information
SP_CTI_FAILURE	Error occurred in external CTI software, see error message for more information

2.4.2.6.3 SP_GETKEYVALUE

Import	Type	Description
HANDLE	SP_HANDLE	Call ID
OBJECT_NAME	SP_IONAM	Object name
OBJECT_NUMBER	SP_IONUM	Object instance
KEYNAME	SP_IOKNAM	Key-value pair name

Export	Type	Description
VALUE	SP_IOKVAL	Key-value pair value

Exceptions	Description
SP_ERROR	General error occurred, see error message for more information
SP_CTI_FAILURE	Error occurred in external CTI software, see error message for more information