CHAPTER 3

Customizing

This chapter discusses how to tailor and use the elements provided as part of integration of Cisco ICM software with Siebel eBusiness applications. It should become obvious that there is no sharp distinction between "configuration" and "customization." However, it seemed useful to separate the choices that need to be made in order to create a working system from the choices that determine precisely how that system works.

This chapter presents information on:

- The definition file CiscoENU.DEF
- Cisco CTI Driver commands and parameters
- Cisco CTI Driver events

Cisco CTI Driver Customization: CiscoENU.DEF

The definition file CiscoENU.DEF (Cisco ENglish USA) controls how the Siebel Client handles CTI events and generates CTI commands. After installing the Cisco CTI Driver for Siebel, this file can be found in (assuming that the default installation directory was used):

c:\Program Files\Cisco\CTI Desktop\CiscoENU.DEF

Understanding CiscoENU.DEF

The information provided here is designed to help understand what action the Cisco CTI Driver for Siebel takes and how it differs from Siebel customization specific CiscoENU.DEF file entries. (For more information regarding Siebel specific customization, see the "Customizing CiscoENU.DEF" section on page 3-8.)

The Cisco CTI Driver for Siebel generates CTI Server events to the Siebel application. These events are explained in the "Cisco CTI Driver Commands and Parameters" section on page 3-14 and the "Cisco CTI Driver Events and Event Data" section on page 3-33. With each event, the driver passes all known call information provided by the CTI Server with each message. When the Siebel application receives an event, it uses the CiscoENU.DEF file to perform a three-step process that maps a CTI Server event to an action in the Siebel application.

Mapping CTI Server Events to the Siebel Application

When the Siebel application receives an event, first it runs the [EventHandler:...] entries in the CiscoENU.DEF file. The EventHandler results in an [EventResponse:...]. After this is run, the driver goes to the [EventLog:...] entries.

This process is explained in more detail below.

EventHandler

The first step starts with entries that begin with [EventHandler: …]. Each of these entries contains DeviceEvent, Order, Filter, Response lines. When the CTI Driver generates an EventAnswer event, the Siebel application will:

1. Only look at EventHandler entries that has DeviceEvent="EventAnswer"
2. Process those entries in the order defined by the Order keyword
3. Test the Filter for a match
4. If there is a match, it jumps to the EventResponse: entry defined in the Response line.

The following is a section of the unmodified CiscoENU.DEF file for reference.

Example 3-1 Sample CiscoENU.DEF Section Regarding Events

```
; --- Internal Agent to Agent Call ---

[EventHandler:InsideCallReceivedFromQueue]

DeviceEvent                    ="EventAnswer"

Response                       ="OnInsideCallReceived"

Filter.SiebelCall              ="?*"

Order                          ="1"


; --- Handle inbound customer call ---

[EventHandler:InboundConsumerCall]

DeviceEvent                    ="EventAnswer"

Response                       ="InboundConsumerCall"

Filter.CED                     ="?*"

Order                          ="2"


[EventResponse:InboundConsumerCall]

QueryBusObj                    ="Consumer"

QueryBusComp                   ="Consumer"

QuerySpec                      ="[CSN]='{CED}'"

SingleView                     ="Consumer Detail View"

FindDialog                     ="Consumer"
```

```
FindField.CSN                    ="{CED}"

SingleLog                        ="LogIncomingCallConsumerFound"

Log                              ="LogIncomingCallConsumerNotFound"


[EventLog:LogIncomingCallConsumerFound]

Display                          ="TRUE"

BusObj                           ="Action"

BusComp                          ="Action"

LogField.Type                    ="Call - Inbound"

LogField.Description             ="Inbound Consumer Call"

LogField.Comment                 ="CSN: {CED}, Dialed Number:
                                 {DialedNumber}"

AfterCall.'ACD Call Duration'    ="{@CallDuration}"


[EventLog:LogIncomingCallConsumerNotFound]

BusObj                           ="Action"

BusComp                          ="Action"

LogField.Type                    ="Call - Inbound"

LogField.Description             ="Unknown Consumer CSN({CED})"

LogField.Comment                 ="CSN: {CED}, Dialed Number:
                                 {DialedNumber}, ANI: {ANI}, DNIS:
                                 {DNIS}"

AfterCall.'ACD Call Duration'    ="{@CallDuration}"
```

In the example above, the first EventHandler entry states when an Answer event is received from the driver, go to the EventResponse:OnInsideCallReceived entry if there is any value in the Filter.SiebelCall field. The second EventHandler entry causes the next step in the mapping process to go to the EventResponse:InboundConsumerCall entry if there is any value in the Filter.CED field.

EventResponse

The second step starts with [EventResponse: …] entries. The response entry performs a database lookup. It also defines the Siebel business object on which to perform a query (QueryBusObj), the query specification (QuerySpec), a view to display, and the action to take if the entry queried is found or not (SingleLog and Log).

In the example shown, the Siebel application will:

1. Perform a lookup on CED, using [CSN]={CED}. The {CED} syntax identifies CED as a field passed by the driver. [CSN] identifies the field as a business object field.
2. If there is a match, the Siebel application will display the view identified by the SingleView entry.
3. If the entry is not found, the Siebel application will pop up the dialog identified by the FindDialog entry.
4. If the database lookup succeeded, it will log an entry identified in the SingleLog line. Otherwise, it will log the entry identified by the Log entry.

EventLog

The third step starts with [EventLog: …] entries. It defines whether to display an error and what to log.

Commands

The CiscoENU.DEF file also defines the action command buttons must execute. A portion is reproduced below, showing how Login works. It maps the toolbar Login button to the Device command Login. It also defines several parameters to pass with the login command, including which ECC variables to register, the agent's skill groups, and agent id, password, extension, and instrument.

Example 3-2 Sample CiscoENU.DEF Section Regarding Commands

```
[Command:Login]

ToolbarCommand                    ="Login "

DeviceCommand                     ="Login"

Description                       ="Agent Login"

Title                             ="&Login"

HotKey                            ="F2"

CmdData                           ="LogInData"


[CmdData:LogInData]

SelectParam                       ="TRUE"

SelectBusObj                      ="CTI Teleset Administration"

SelectBusComp                     ="CTI Teleset Administration"

SelectApplet                      ="Default Pick Applet"

SelectTitle                       ="Select an Extension:"

Param.AgentID                     ="{@AgentId}"

Param.Password                    ="{@AgentPin}"
```

```
Param.Extension                    ="[Extension :Lookup]"

Param.Instrument                   ="[Extension :Lookup]"


Param.NumECCVars                   ="8"

Param.RegECCVar1                   ="Consumer_Id"

Param.RegECCVar2                   ="CampID"

Param.RegECCVar3                   ="CampContactID"

Param.RegECCVar4                   ="RowID"

Param.RegECCVar5                   ="Contact_Account_Id"

Param.RegECCVar6                   ="Contact_Id"

Param.RegECCVar7                   ="ContactPhone"

Param.RegECCVar8                   ="Id"

Param.SkillGroupNumber1            ="5000"

Param.SkillGroupName1              ="service"

Param.SkillPriority1               ="0"

Param.NumSkillGroups               ="1"
```

Note ECC and skill group information is optional.

It should be clear that there are a great number of special fields, specific values, particular views, and specific database schema. Each Siebel customer customizes the database and business objects to suit their business needs. The degree of knowledge required is very high and requires a Siebel Certified Engineer.

Customizing CiscoENU.DEF

CiscoENU.DEF, as provided, is only a sample file. It will *not* work until you make appropriate modifications to CiscoENU.DEF and/or to Siebel Configuration Parameters through Siebel Call Center Administration. The idea is to add filter information to the CiscoENU.DEF file in order to trigger the appropriate response by the Siebel application when a call is received. The filter operates on information provided with the CTI Server message, together with other information tracked by driver.

Note The CiscoENU.DEF file, as provided, is intended to work with the demo described in Chapter 5, "Demonstrating."

The following sections provide illustrative examples of what can be done by editing the out-of-the-box CiscoENU.DEF.

LogIn Command

Usually the Siebel Client is configured to automatically log in the user who is launching the application. However, there may be situations where this is not the appropriate behavior; for example, where more than one person uses the same workstation. In such a situation, the LogIn command can be reconfigured to ask the user for an ID and password. In the following example, the user is also asked to provide information regarding the telephone that will be used.

Warning **If configured for autologin, do not also use an applet to prompt for login ID and password. In autologin mode, the applet window will not display correctly and you cannot select user ID or password. It will also lock autologin from working. Use the applet only when autologin is not enabled.**

Compare the example below to the information given about the LogIn command in the "Cisco CTI Driver Commands and Parameters" section on page 3-14.

Note Extension and Instrument are optional in the Login command. If no extension or instrument are passed, the Cisco CTI Driver for Siebel will default to the extension passed in the Teleset (DN1).

```
[Command:Login]

DeviceCommand             = "LogIn"

CmdData                   = "LogInData"

Title                     = "Log &In"

Hidden                    = "TRUE"

Order                     = 2


[CmdData:LogInData]

SelectParam               = "TRUE"

SelectBusObj              = "CTI Teleset Administration"

SelectBusComp             = "CTI Teleset Administration"

SelectApplet              = "Default Pick Applet"

SelectTitle               = "Select an Extension:"

Param.AgentID             = "{@AgentId}"

Param.Password            = "{@AgentPin}"

Param.Extension           = "[Extension]"

Param.Instrument          = "[Extension]"

Param.NumSkillGroups      = "1"

Param.SkillGroupNumber1   = "5000"

Param.SkillGroupName1     = "service"

Param.SkillPriority1      = "0"
```

```
Param.NumECCVars          = "8"

Param.RegECCVar1          = "Consumer_Id"

Param.RegECCVar2          = "CampID"

Param.RegECCVar3          = "CampContactID"

Param.RegECCVar4          = "RowID"

Param.RegECCVar5          = "Contact_Account_Id"

Param.RegECCVar6          = "Contact_Id"

Param.RegECCVar7          = "ContactPhone"

Param.RegECCVar8          = "Id"

Param.UserID              =”{@UserName}”
```

Note Skill group information is optional, except for the Alcatel switch. If no skill group information is to be specified, remove any entries for Param.NumSkillGroups, Param.SkillGroupNumber, Param.SkillGroupName, and Param.SkillPriority. The skill group information is required for the Alcatel switch.

Screen Pop

As an incoming call is connected to an agent, it is usually desirable to display information about the call to the agent. The agent's screen can display such information, retrieved from a database, provided that appropriate selection criteria are met. In the following example, the screen is populated with information about the caller if the CallVariable1 field from the incoming call equals the Work Phone Number of a record existing in the Contact object.

For general information on events, see the “Cisco CTI Driver Events and Event Data” section on page 3-33.

```
[EventHandler:InboundCallReceived]

DeviceEvent                = "EventAnswer"

Response                   = "OnInboundCallReceived"

Filter.CallVariable1       = "*"

Order                      = "8"


[EventResponse:OnInboundCallReceived]

QueryBusObj                = "Contact"

QueryBusComp               = "Contact"

QuerySpec                  = "'Work Phone #'='{CallVariable1}'"

SingleView                 = "Service Contact Detail View"

MultiView                  = "Contact List View"

FindDialog                 = "Service Request"

FindField.CSN              = "Ask Caller"

SingleLog                  = "LogIncomingCallContactFound"

Log                        = "LogIncomingCallContactNotFound"
```

In the above case, if there was no match between CallVariable1 and Work Phone #, a search screen appears allowing the agent to do a search. While talking to the caller, the agent may perform a search or may simply create a new record.

Simultaneously, for tracking purposes, the call is logged—even if the caller is unknown. To do this on the Event response, the following scripts are invoked:

```
[EventLog:LogIncomingCallContactFound]

Display                    = "TRUE"

BusObj                     = "Action"

BusComp                    = "Action"

LogField.Type              = "Call - Inbound"

LogField.Description       = "Call - Inbound"

LogField.Comment           = "Account: {CallVariable1}
                           {Contact.Account}"

AfterCall.'ACD Call Duration'= "{@CallDuration}"



[EventLog:LogIncomingCallContactNotFound]

BusObj                     = "Action"

BusComp                    = "Action"

LogField.Type              = "Call - Inbound"

LogField.Description       = "Unknown Caller
                           ({CallVariable1})"

LogField.Comment           = "Call Id: {CallID}, ANI: {ANI},
                           CED: {CED}, DNIS: {DNIS}"

AfterCall.'ACD Call Duration'= "{@CallDuration}"
```

Table 3-1 describes the commands available to configure the CiscoENU.DEF file to do a screen pop.

Table 3-1 CiscoENU.DEF Screen Pop Commands

Command	Description
ViewCallObject	Enables viewing of call and Siebel event details using the ViewCall Siebel CTI menu option.
Associate	Enables screen-pop transfer using the Siebel call-tracking object. To create this call object, use the logging feature, [EventLog], and the AfterCall event log parameter.

Invoking a Script

The following example illustrates how you might invoke a Siebel script.

```
[EventHandler:InboundCallReceived]

DeviceEvent               ="EventAnswer"

Filter.CallVariable1      ="*"

Order ="8"

Response                  ="OnInboundCallReceived"


[EventResponse:OnInboundCallReceived]

Script                    =ProcEventAnswer

FindDialog                ="Service Request"

FindField.CSN             ="Ask Caller"

Log                       ="LogIncomingCallContactNotFound"

MultiView                 ="Contact List View"

QueryBusComp              ="Contact"
```

```
QueryBusObj              ="Contact"

QuerySpec                ="'Work Phone #'='{CallVariable1}'"

SingleLog                ="LogIncomingCallContactFound"

SingleView               ="Service Contact Detail View"

UseCtxData               ="TRUE"
```

Enabling Telephone Status Functionality

The Call Center administrator is able to track all agent states, call states and attributes such as TalkingTime and CallDuration in real time using Telephone Status View from the Siebel window. In order for this functionality to work, the following needs to be included in the Cisco.ini file.

```
[Setting]

UpdatePhoneStatusTable   =”TRUE”
```

Cisco CTI Driver Commands and Parameters

Table 3-2 lists the Cisco CTI Driver commands. A section listing corresponding DEF file mapping samples for some switches follows this table.

Table 3-3 defines the parameters associated with these commands.

Note The Siebel “CTI Not Ready State” button maps to both Available/Ready and NotReady agent states. When this button is enabled, the agent is in the NotReady state; when the button is disabled, the agent is Available/Ready. Similarly, the Siebel “CTI Busy State” button maps to both WorkReady and WorkNotReady. When enabled, the agent state is WorkReady; when disabled, the agent state is WorkNotReady.

Table 3-2 Cisco CTI Driver for Siebel Commands

Command	Parameters	Description
AnswerCall		Answer the first incoming call
ChangeBusyState	ReasonCode	Toggle agent state between Ready and NotReady **Note** The Meridian switch does not support the NotReady state, so it will not toggle between Ready and NotReady.
ChangeReadyState	ReasonCode	Change agent state to Ready only
ChangeNotReadyState	ReasonCode	Change agent state to NotReady only **Note** The Meridian switch does not support the NotReady state.

Table 3-2 Cisco CTI Driver for Siebel Commands (continued)

Command	Parameters	Description
ChangeWorkReadyState	ReasonCode	Toggle agent state between WorkReady and NotReady **Note** For Avaya and Meridian switches, the agent can use the SetWorkReadyState button to go to the WorkReady state after releasing the call. For example, after an agent has finished a call, instead of being placed in the Ready state, the agent chooses to be unavailable for another call by pressing the SetWorkReadyState button while on a call, which places the agent in the WorkReady state after the call is finished. Observe the following switch restrictions: • Aspect: Does not support the WorkReady state, so it will not toggle between WorkReady and NotReady, and will only change agent state to NotReady • Avaya: Will change from WorkReady to NotReady, but will not toggle back to WorkReady • Meridian: Does not support the NotReady state, so it will not toggle between WorkReady and NotReady, and will only change agent state to WorkReady **Note** For Enterprise Agent, this changes WrapUp state.

Table 3-2 Cisco CTI Driver for Siebel Commands (continued)

Command	Parameters	Description
ChangeWrapUpState	ReasonCode	On Aspect switches only, drops current call and changes agent state to NotReady **Note** This command is currently supported only on Aspect switches and is only enabled if there is an active (not held) call; otherwise, the switch displays an error.
ConferenceComplete		Complete the consultative conference
ConferenceInit	CallVariable1 … CallVariable10 DialedNumber UUI CallWrapUp FacilityCode AuthorizationCode AccountCode CallPlacementType CallManner FacilityType AlertRings CallOption PostRoute NumECCVars ECC.*RegECCVarName1* … ECC.*RegECCVarNameN*	Begin consultative conference—the caller is put on hold, and the current agent dials another agent's extension
HoldCall		Hold current active call

Table 3-2 Cisco CTI Driver for Siebel Commands (continued)

Command	Parameters	Description
LogIn	AgentID Extension Instrument Password CallVariableMask NumECCVars RegECCVar1 … RegECCVarN NumSkillGroups SkillGroupNumber1 SkillGroupName1 SkillPriority1 … SkillGroupNumberN SkillGroupNameN SkillPriorityN ReasonCode	Agent login
LogOut	ReasonCode	Agent logout
MakeCall	CallVariable1 … CallVariable10 DialedNumber UUI CallWrapUp FacilityCode AuthorizationCode AccountCode CallPlacementType CallManner FacilityType AlertRings CallOption PostRoute NumECCVars ECC.*RegECCVarName1* … ECC.*RegECCVarNameN*	Make the outbound call
ReleaseCall		Release current call

Table 3-2 Cisco CTI Driver for Siebel Commands (continued)

Command	Parameters	Description
RetrieveCall		Retrieve the original call after ConferenceInit or TransferInit RetrieveCall is an Alternate command that allows you to switch between active calls and other calls. For example: 1. Agent answers a call and clicks the Retrieve button. 2. Agent transfers to another agent. 3. Another agent answers a consultation call. 4. Consultation call is held and the original call is retrieved. **Note** This behavior only applies to the Alcatel, Aspect, Avaya, IPCC, and Spectrum switches. The Meridian switch does not support the Alternate/Switch command and uses the Reconnect command instead, and releases an active call before activating another one.
SimulateCall	CallVariable1	Simulate incoming call
TransferComplete		Complete the consultative transfer

Table 3-2 Cisco CTI Driver for Siebel Commands (continued)

Command	Parameters	Description
TransferInit	CallVariable1 … CallVariable10 DialedNumber UUI CallWrapUp FacilityCode AuthorizationCode AccountCode CallPlacementType CallManner FacilityType AlertRings CallOption PostRoute NumECCVars ECC.*RegECCVarName1* … ECC.*RegECCVarNameN*	Begin consultative transfer— the caller is put on hold, and the current agent dials another agent's extension
TransferMute	CallVariable1 … CallVariable10 DialedNumber UUI CallWrapUp FacilityCode AuthorizationCode AccountCode CallPlacementType CallManner FacilityType AlertRings CallOption PostRoute NumECCVars ECC.*RegECCVarName1* … ECC.*RegECCVarNameN*	Make blind (single-step) transfer of the caller

Table 3-2 Cisco CTI Driver for Siebel Commands (continued)

Command	Parameters	Description
UnHoldCall		Take current call off hold
UpdateCurCallData	CallVariable1 … CallVariable10 CallWrapUp DialedNumber CED ECC.*RegECCVarName1* … ECC.*RegECCVarNameN*	Update the current call information. It is recommended that this command only be invoked from within a Siebel VB or eSiebel script. See the example in the "Example of a Siebel VB Script" section on page 3-31. **Note** The name UpdateCurCallData was chosen over AttachData to prevent any confusion with the AttachContext parameter used in all event handler mappings that require Siebel Manager to generate and return a View Bookmark for the current screen on display.

File Mapping Samples

The following sections list corresponding DEF file mapping samples for the Aspect, Avaya, IPCC, and Meridian switches.

Aspect

The following example illustrates mapping by pairs using ChangeReadyState and ChangeNotReadyState.

Note Another example of switching between Ready and NotReady is to only use one button, then map only one Siebel button to ChangeBusyState Device command.

Example 3-3 Sample Using ChangeReadyState and ChangeNotReadyState

```
[Command:ChangeBusyState]

Description                          ="Switch To Ready State"

DeviceCommand                        ="ChangeReadyState"

Hidden                               ="FALSE"

Order                                ="1"

ToolbarCommand                       ="ChangeBusyState"


[Command:ChangeNotReadyState]

Description                          ="Switch To NotReady State"

DeviceCommand                        ="ChangeNotReadyState"

Hidden                               ="TRUE"

Order                                ="1"

ToolbarCommand                       ="ChangeNotReadyState";
```

The following example illustrates mapping by pairs using ChangeBusyState and ChangeNotReadyState.

Example 3-4 Sample Using ChangeBusyState and ChangeNotReadyState

```
[Command:ChangeBusyState]

Description                          ="Switch To ReadyNotReady State"

DeviceCommand                        ="ChangeBusyState"

Hidden                               ="TRUE"
```

```
HotKey                          ="F9"

Order                           ="1"

Title                           ="Ready/NotReady &State"

ToolbarCommand                  ="ChangeBusyState"


[Command:ChangeNotReadyState]

Description                     ="Switch To WrapUp State"

DeviceCommand                   ="ChangeWrapUpState"

Hidden                          ="TRUE"

HotKey                          ="F10"

Order                           ="1"

Title                           ="&Not Ready State"

ToolbarCommand                  ="ChangeNotReadyState";
```

- The ChangeBusyState button then toggles between Ready and NotReady.
- The ChangeNotReadyState button is available if there is an active call, and then brings the agent to a WrapUp state.

Avaya

The following example illustrates mapping by pairs using ChangeReadyState and ChangeNotReadyState (if WorkReady is never used explicitly).

Note Another example of switching between Ready and NotReady (not using WorkReady) is to only use one button, then map only one Siebel button to ChangeBusyState Device command.

Example 3-5 Sample Using ChangeReadyState and ChangeNotReadyState

```
[Command:ChangeBusyState]

Description                          ="Switch To Ready State"

DeviceCommand                        ="ChangeReadyState"

Hidden                               ="FALSE"

Order                                ="1"

ToolbarCommand                       ="ChangeBusyState"


[Command:ChangeNotReadyState]

Description                          ="Switch To NotReady State"

DeviceCommand                        ="ChangeNotReadyState"

Hidden                               ="TRUE"

Order                                ="1"

ToolbarCommand                       ="ChangeNotReadyState";
```

IPCC

The following example illustrates mapping by pairs using ChangeBusyState and ChangeWorkReadyState.

Example 3-6 Sample Using ChangeBusyState and ChangeWorkReadyState

```
[Command:ChangeBusyState]

Description                          ="Switch To AfterCall WrapUp
                                     State"
```

```
DeviceCommand                    ="ChangeWorkReadyState"

Hidden                           ="FALSE"

Order                            ="1"

ToolbarCommand                   ="ChangeBusyState"


[Command:ChangeNotReadyState]

Description                      ="Switch To Ready/NotReady State"

DeviceCommand                    ="ChangeBusyState"

Hidden                           ="TRUE"

Order                            ="1"

ToolbarCommand                   ="ChangeNotReadyState";
```

Meridian

The following example illustrates mapping by pairs using ChangeBusyState and ChangeWorkReadyState.

Note Another example of switching between Ready and WorkReady is to only use one button, then map only one Siebel button to ChangeBusyState Device command.

Example 3-7 Sample Using ChangeBusyState and ChangeWorkReadyState

```
[Command:ChangeBusyState]

Description                      ="Switch To WorkReady State"

DeviceCommand                    ="ChangeWorkReadyState"
```

```
Hidden                                  ="FALSE"

Order                                   ="1"

ToolbarCommand                          ="ChangeBusyState"


[Command:ChangeNotReadyState]

Description                             ="Switch To Ready/WorkReady
                                        State"

DeviceCommand                           ="ChangeBusyState"

Hidden                                  ="TRUE"

Order                                   ="1"

ToolbarCommand                          ="ChangeNotReadyState";
```

Table 3-3 Cisco CTI Driver Command Parameters

Command Parameter	Type	Command Parameter Description
AccountCode	String	A cost-accounting or client number used by the peripheral for charge-back purposes.
AgentID	String	The agent's ACD login ID.
AlertRings	Number	The maximum rings that the call's destination will remain alerting. Zero indicates that the peripheral default (typically 10 rings) should be used.
AuthorizationCode	String	The authorization code needed to access the resources required to initiate the call.
CallManner	String	Specifies additional call processing options. See Table 3-4.
CallOption	String	Specifies additional peripheral-specific call options. See Table 3-5.
CallPlacementType	String	Specifies how a call is to be placed. See Table 3-6.

Table 3-3 Cisco CTI Driver Command Parameters (continued)

Command Parameter	Type	Command Parameter Description
CallVariable1 … CallVariable10	String	Call-related variable data.
CallVariableMask	Number	A bitwise combination of CallVariable masks (see Table 3-7) corresponding to the call variables that the client wishes to receive.
CallWrapUp	String	Call-related wrap-up data. **Note** The agent has a maximum of two minutes in which to enter wrap-up data.
CED	String	Caller Entered Digits.
DialedNumber	String	Phone number (dialing pattern). Dial:FilterRule configuration parameters must be defined that filter full telephone numbers from a database so they can be handled as extensions
ECC.*RegECCVarName1* … ECC.*RegECCVarNameN*	String	The value contained in the named Expanded Call Context variable (see RegECCVar below). Expanded Call Context variables may consist of an arbitrary number of named variable fields, subject only to a combined total limit of 2000 bytes. These variables are available only if the Expanded Call Context feature is explicitly enabled in ICM software. For specifics on enabling, creating, and naming the ECC variables, see the *Cisco ICM Software Script Editor Guide*. As an example, the command parameters associated with the LogInData ECC variables given in Section would be: ECC.Consumer_Id ECC.CampID … ECC.Id
Extension	Number	The agent's ACD teleset extension.
FacilityCode	String	A trunk access code, split extension, or other data needed to access the chosen facility.
FacilityType	String	The type of facility to be used. See Table 3-8.
Instrument	Number	The agent's ACD instrument number.

Table 3-3 Cisco CTI Driver Command Parameters (continued)

Command Parameter	Type	Command Parameter Description
NumECCVars	Number	The number of registered Expanded Call Context named variables.
NumSkillGroups	Number	The number of Skill Groups (which is also the number of Skill Group Names, of Skill Group Numbers, and of Skill Group Priorities) that the agent is currently associated with.
Password	String	The agent's ACD password.
PostRoute	String (TRUE or FALSE)	When this field is set to TRUE and a DialedNumber is provided instead of a held call (blind transfer), the Post-Routing capabilities of Cisco ICM software are to be used to determine the new call destination.
ReasonCode	Number	A peripheral-specific code indicating the reason for logging out.
RegECCVar1 … RegECCVarN	String	The registered name of an Expanded Call Context named variable.
SkillGroupName1 … SkillGroupNameN	String	The name of an agent SkillGroup queue.
SkillGroupNumber1 … SkillGroupNumberN	Number	The number of an agent SkillGroup queue.
SkillPriority1 … SkillPriorityN	Number	The priority of the skill group, or 0 when skill group priority is not applicable or not available.
UUI	String	The ISDN user-to-user information.

The following tables list the values associated with certain parameters described in Table 3-3.

CallManner Values

Table 3-4 shows the possible CallManner values.

Table 3-4 CallManner Values

CallManner	Description
CmtUnspecified	Use default call manner.
CmtPolite	The call should only be attempted if the originating device is idle.
CmtBelligerent	The call should always be attempted, disconnecting any currently active call.
CmtSemiPolite	The call should only be attempted if the originating device is idle or is receiving dial tone.

CallOption Values

Table 3-5 shows the possible CallOption values.

Table 3-5 CallOption Values

CallOption	Description
COptUnspecified	No call options specified, use defaults.
COptCallingAgentOnline	Attempt the call only if the calling agent is “online” (available to interact with the destination party).
COptCallingAgentReserved	Attempt the call only if ACDNR on the calling agent’s set is activated (DMS-100).
COptCallingAgentNotReserved	Attempt the call only if ACDNR on the calling agent’s set is not activated (DMS-100).
COptCallingAgentBuzzBase	Causes a buzz to be applied to the base of the telephone set as the call is initiated (DMS-100).

Table 3-5 CallOption Values (continued)

CallOption	Description
COptCallingAgentBeepHset	Causes a tone to be applied to the agent headset as the call is initiated (DMS-100).
COptServiceCircuitOn	Causes a call classifier to be applied to the call (DEFINITY ECS).

CallPlacementType Values

Table 3-6 shows the possible CallPlacementType values.

Table 3-6 CallPlacementType Values

CallPlacementType	Description
CptUnspecified	Use default call placement
CptLineCall	An inside line call
CptOutbound	An outbound call
CptOutboundNoAccessCode	An outbound call that will not require an access code
CptDirectPosition	A call placed directly to a specific position
CptDirectAgent	A call placed directly to a specific agent
CptSupervisorAssist	A call placed to a supervisor for call handling assistance

CallVariable Masks

Table 3-7 lists the CallVariable masks.

Table 3-7 CallVariable Masks

Mask Name	Description	Value
CALL_VAR_1_MASK	CallVariable1	0x0001
CALL_VAR_2_MASK	CallVariable2	0x0002
CALL_VAR_3_MASK	CallVariable3	0x0004
CALL_VAR_4_MASK	CallVariable4	0x0008
CALL_VAR_5_MASK	CallVariable5	0x0010
CALL_VAR_6_MASK	CallVariable6	0x0020
CALL_VAR_7_MASK	CallVariable7	0x0040
CALL_VAR_8_MASK	CallVariable8	0x0080
CALL_VAR_9_MASK	CallVariable9	0x0100
CALL_VAR_10_MASK	CallVariable10	0x0200

FacilityType Values

Table 3-8 shows the possible FacilityType values.

Table 3-8 FacilityType Values

FacilityType	Description
FtUnspecified	Use default facility type.
FtTrunkGroup	Facility is a trunk group.
FtSkillGroup	Facility is a skill group or split.

Example of a Siebel VB Script

This example illustrates the use of the Cisco CTI Driver command UpdateCurCallData. For invoking a script, see the "Invoking a Script" section on page 3-13.

```
Declare Sub ShowCallData(ctiCallData as CTIData)

Function ProcEventAnswer() as Integer

    dim ctiService as CTIService
    dim ctiEventParam as CTIData
    dim ctiUpdatedEventParam as CTIData

    set ctiService = TheApplication.GetCTIService

    set ctiEventParam = ctiService.GetCurrentCallData

    Call ShowCallData(ctiEventParam)

    set ctiUpdatedEventParam = ctiService.CreateData

    ctiUpdatedEventParam.SetFieldValue "ECC.user.bobc","Changed in Siebel VB Script..."

    ctiService.InvokeCommandWithData "UpdateCurCallData",ctiUpdatedEventParam

    ProcEventAnswer = ContinueOperation

End Function

Sub ShowCallData( ctiCallData as CTIData)

    dim nParamCount as integer
    dim strParamList as string
    dim strParamName as string
    dim strParamValue as string
    dim nI as Integer

    strParamList =""
    nParamCount = ctiCallData.GetCount

    if nParamCount > 0 Then
        for nI=0 to (nParamCount - 1)
            strParamName  = ctiCallData.GetFieldAt(nI)
            strParamValue = ctiCallData.GetFieldValue(strParamName)
            strParamList = strParamList & "ctiCallData(" & nI & ")=" & "<" & strParamName
                    & "," & strParamValue & ">" & chr(13)
        next nI
    end if

    MsgBox strParamList
End Sub
```

Cisco CTI Driver Events and Event Data

Table 3-9 lists the Cisco CTI Driver events. A section illustrating the EventTransferred floating part follows this table.

Table 3-10 lists the data associated with the events.

Table 3-11 lists how the Cisco CTI Driver events map to the CTI Server events.

Table 3-9 Cisco CTI Driver Events

Event Name	Event Description
EventAgentLogin	Agent logged into ACD
EventAgentLogout	Agent logged out from ACD
EventAgentNotReady	Agent state set to NotReady
EventAgentReady	Agent state set to Ready
EventAnswer	Inbound call answered
EventConferenced	Conference was completed successfully
EventControlError	Control error occurred. This event indicates that the Cisco CTI Driver has received a CONTROL_FAILURE_CONF or CONTROL_FAILURE_EVENT message from the CTI Server. The parameters associated with this event are: ActionCode. Describes the action requested of the CTI Server: the values are found in the MethodType Values table in the *Cisco ICM Software Desktop Control Server Reference Guide.* ControlFailureCode. The values are found in the ControlFailureCode Values table in the *Cisco ICM Software Desktop Control Server Reference Guide.*
EventDialing	Dialing digits

Table 3-9 Cisco CTI Driver Events (continued)

Event Name	Event Description
EventError	Error occurred. This event indicates that the Cisco CTI Driver has received a FAILURE_CONF or FAILURE_EVENT message from the CTI Server. The parameters associated with this event are: ActionCode. Describes the action requested of the CTI Server: the values are found in the MethodType Values table in the *Cisco ICM Software Desktop Control Server Reference Guide.* StatusCode. The values are found in the Status Code table in the *Cisco ICM Software Desktop Control Server Reference Guide.*
EventEstablished	Outgoing call answered
EventReleased	Call released
EventRetrieved	Original call was retrieved
EventRinging	Incoming call ringing
EventServerOffline	Cisco CTI Server was disconnected
EventTransferred	Call was transferred successfully This event contains a floating part, which arrives at the originator and the destination agent's desktop with the following configurations: • PrimaryDeviceID=<*Primary Device ID*> • SecondaryDeviceID=<*Secondary Device ID*> • TransferringDeviceID=<*Transferring Device ID*> • TransferredDeviceID=<*Transferred Device ID*> • ConnectedPartyCallID1=<*Connected Party call ID1*> • ConnectedPartyCallID2=<*Connected Party call ID2*> • ConnectedPartyCallID3=<*Connected Party call ID3*> • ConnectedPartyCallID4=<*Connected Party call ID4*>

Floating Part

The following example of a .DEF file verifies that a floating part arrives along with an event, and illustrates the values for {'PrimaryDeviceID'}, {'SecondaryDeviceID'}, {'TransferringDeviceID'}, and {'ConnectedPartyCallID1'}, which display on the Siebel FindDialog "Corporate Contact", if event data (in this case, call data) meet all filter conditions on EventHandler:TransferComplete.

Note The sample is for instructional purposes only.

Example 3-8 Sample Using a Floating Part of DeviceEvent EventTransferred

```
[EventHandler:TransferComplete]

DeviceEvent                          ="EventTransferred"

Filter.PrimaryDeviceID               ="23817"

Filter.SecondaryDeviceID             ="23817"

Filter.TransferredDeviceID           ="23818"

Filter.TransferringDeviceID          ="23817"

Order                                ="1"

Response                             ="OnEventTransferredReceived"


[EventLog:LogTransferCompleteNotFound]

BusComp                              ="Action"

BusObj                               ="Service Request"

Command                              ="AssociateContact"
```

```
Display                                 ="TRUE"

LogField.'Account Id'                   ="{'Account Id'}"

LogField.'Activity SR Id'               ="{'Account Id'}"

LogField.'SR Number'                    ="{'Account Id'}"

LogField.Comment                        ="Call Transfered received to
                                        ........========="

LogField.Description                    ="sample description"


[EventLog:LogTransferCompleteFound]

BusComp                                 ="Action"

BusObj                                  ="Service Request"

Command                                 ="AssociateContact"

Display                                 ="TRUE"

LogField.'Account Id'                   ="{'Account Id'}"

LogField.'Activity SR Id'               ="{'Account Id'}"

LogField.'SR Number'                    ="{'Account Id'}"

LogField.Comment                        ="Call Transfered received to
                                        ........========="

LogField.Description                    ="sample description"


[EventResponse:OnEventTransferredReceived]

FindDialog                              ="Corporate Contact"

FindField.CSN                           ="{'SecondaryDeviceID'}"
```

```
FindField.Last Name                    ="{'TransferringDeviceID'}"

Log                                    ="LogTransferCompleteNotFound"

MultiView                              ="Contact List View"

QueryBusComp                           ="Contact"

QueryBusObj                            ="Contact"

QuerySpec="'CSN'                       ='{PrimaryDeviceID}'"

SingleLog                              ="LogTransferCompleteFound"

SingleView                             ="Service Contact Detail View"

UseCtxData                             ="TRUE"

FindField.First Name                   ="{'ConnectedPartyCallID1'}"
```

Table 3-10 Cisco CTI Driver Event Data

Event Field Name	Type	Event Field Description
ANI	String	Automatic Number Identification
CallID	String	Call identifier
CallType	String	The general classification of the call type. See Table 3-12.
CallVariable1 … CallVariable10	String	Call-related variable data
CallWrapUp	String	Call-related wrap-up data
CED	String	Caller Entered Digits in response to IVR prompting
DialedNumber	String	The number dialed
DNIS	String	Dialed Number Identification Service

Table 3-10 Cisco CTI Driver Event Data (continued)

Event Field Name	Type	Event Field Description
ECC.*RegECCVarName1* … ECC.*RegECCVarNameN*	String	The value contained in the named Expanded Call Context variable. Expanded Call Context variables may consist of an arbitrary number of named variable fields, subject only to a combined total limit of 2000 bytes. These variables are available only if the Expanded Call Context feature is explicitly enabled in ICM software. For specifics on enabling, creating, and naming the ECC variables, see the *Cisco ICM Software Script Editor Guide.*
LineType	String	The general classification of the line type. See Table 3-13.
UUI	String	The ISDN user-to-user information

Table 3-11 Cisco CTI Driver and CTI Server Events

Driver Event	CTI Server Event
EventDialing	CALL_ORIGINATED_EVENT
EventRinging	CALL_DELIVERED_EVENT
EventAnswer	CALL_ESTABLISHED_EVENT
EventRetrieved	CALL_RETRIEVED_EVENT
EventReleased	CALL_CLEARED_EVENT
EventCallDataUpdated	CALL_DATA_UPDATE_EVENT
EventTransferred	CALL_TRANSFERRED_EVENT
EventConferenced	CALL_CONFERENCED_EVENT
EventAgentLogin	AGENT_STATE_EVENT and new agent state is Login
EventAgentLogout	AGENT_STATE_EVENT and new agent state is Logout
EventAgentReady	AGENT_STATE_EVENT and new agent state is Ready
EventAgentNotReady	AGENT_STATE_EVENT and new agent state is Not Ready
EventAgentBusy	AGENT_STATE_EVENT and new agent state is Busy

Table 3-11 Cisco CTI Driver and CTI Server Events (continued)

Driver Event	CTI Server Event
EventAgentNotBusy	AGENT_STATE_EVENT and new agent state is Not Busy
EventServerUnavailable	Connection to CTI Server broken
EventError	Error generated on last request
EventUserMessage	USER_MESSAGE_EVENT

CallTypes

Call types were mentioned in Table 3-10.

Table 3-12 shows the possible CallTypes.

Table 3-12 CallType Values

Call Type	Description
ACDIn	Inbound ACD call.
AgentInside	Agent inside call.
AgentOut	Agent out call.
AutoOut	Automatic out call.
Conference	Conference call.
Consult	Consult call.
ConsultConference	Conferenced consult call.
ConsultOffered	Announced transferred call.
Offered	Blind/single-step transferred call.
OtherIn	Inbound call.
Out	Outbound call.
OverflowIn	Overflowed inbound call.
PreRouteACDIn	Translation routed inbound ACD call.
PreRouteDirectAgent	Translation routed call to a specific agent.

Table 3-12 CallType Values (continued)

Call Type	Description
TransferIn	Transferred inbound call.
Unmonitored	Inside or outbound call for which *no* call events will be received.

LineTypes

Line types were mentioned in Table 3-10.

Table 3-13 shows the possible LineTypes.

Table 3-13 LineType Values

Line Type	Description
DID	Direct inward dial call.
Help	Assistance call.
InboundACD	Inbound ACD call.
Inside	Inside call.
Message	Voice message call.
OutboundACD	Outbound ACD call.
Supervisor	Supervisor call.
Unknown	Any purpose call.

Switch-Specific Comments

Cisco CTI Driver for Siebel currently supports the following peripherals:

- Alcatel
- Aspect
- Enterprise Agent and IPCC
- Nortel DMS-100
- Nortel Meridian

- Rockwell Spectrum
- Siemens Hicom

In a few cases, there are specific settings that need to be made for particular switches. This section lists those particulars.

Alcatel

- The Alcatel switch is supported by CTI 4.1.6 and above.
- When the Agent login to a device, the device becomes the agent. For example, when agent 3550 logs into device 3300, one would dial 3550 to reach the agent.
- Position ID is a required part of the Login information.
- The Agent must specify a skill group when logging in.
- Single step transfer and conference are not supported; transfer and conference calls must be consultative.
- An inside call cannot be put on Hold.
- A second line is not supported. A second call can only be made as a Consult call in the context of an existing call (via Transfer or Conference).
- An inside call cannot be made to a pilot.

PeripheralType Parameter

A PeripheralType parameter must be included in the Cisco.INI file, namely:

Service:PeripheralType = PTAlcatel

Skill Group Information

Skill Group information must be provided as LoginData in the CiscoENU.DEF file:

NumSkillGroups

SkillGroupNumber1

SkillGroupName1

SkillPriority1

…

SkillGroupNumberN

SkillGroupNameN

SkillPriorityN

Aspect

- Blind Transfer (TransferMute) is supported by this switch at the PIM level, but one-step transfers are not supported on a hard phone.
- The Agent can only logout while in the NotReady state.

ReasonCode

To use a specific reason code, the ReasonCode parameter must be added to the relevant [CmdData:<*AgentStateChange*>] in CiscoENU.DEF. For example:

```
[Command:Logout]

DeviceCommand             = "LogOut"

CmdData                   = "LogOutData"

Hidden                    = "TRUE"

Order                     = "1"

ToolbarCommand            = "LogOut"


[CmdData:LogOutData]

Param.ReasonCode          = "12345"
```

Enterprise Agent and IPCC

- Enterprise Agent and IPCC are only supported by CTI Server Protocol 6.
- LOGOUT and MAKE_CALL are only supported when the agent is in the NotReady state.
- Consult and blind transfers are supported. However, placing a call on hold, making a new call and then completing the transfer is not supported.
- Hold and retrieve are supported, but not during a consult call because it breaks the consult relationship with the Call Manager. AlternateCall is also not supported because it is a hold and a transfer. You are unable to Hold and Retrieve when in the middle of a Transfer or Conference.
- Completing a conference or a transfer to a consulted agent on hold is not supported.
- Overlapping transfer and conference consult operations on the same parties are not supported. For example, Agent A calls Agent B. During the conversation, Agent A needs to conference consult Agent C. Agent B feels that Agent D has more information, so Agent B then transfer consults to Agent D. To end the call, Agent A completes the conference and Agent B completes the transfer. This would fail.
- Only the conference initiator can add parties to the conference.
- Calls do not get queued at the Call Manager but instead at some queue point. Because of this, skill group queue statistics are not available via the QUERY_SKILL_GROUP_STATISTICS_REQ. Service controlled IVRs can be monitored via CTI to get queued and dequeued events, as well as established events.
- A CALL_CONNECTION_CLEARED_EVENT may be received with a cause of CEC_REDIRECTED for the following cases:
  - Agent calls a CTI Route Point and call is directed to another resource
  - Agent calls an IVR and the IVR redirects the call
  - Agent calls a number with a forwarding option turned on
- When an invalid number is dialed, the call will fail and the phone will ring fast busy. The call appearance will remain active until the Call Manager cleans up the call or the agent releases the failed call either via the hard phone or through the agent soft phone. The time out on the Call Manager is about 30 seconds.

- Call Manager Multi-line feature (an agent with more than one monitored ACD line) is not supported.
- Call Manager Shared line feature (agents share the same extension) is not supported.

The agent desktop settings parameters need to be configured in the Cisco.ini file to match the ICM configuration. For example:

```
[Setting]

AutoLogin              ="FALSE"



[Driver]

CtiSimPath             ="c:\program
                       files\Geotel\Tools\CtiSim.exe"

CtiSimPort             ="199"

IsCtiSim               ="TRUE"

LogFileName            ="C:\CTISim"



[Service]

PeripheralID           ="1"

SideAHost              ="localhost"

SideAPort              ="199"

SideBHost              ="localhost"

SideBPort              ="199"

WrapupInMode           ="OPTIONAL"

WrapupOutMode          ="OPTIONAL"
```

WrapupInMode is the wrapup mode variable for incoming calls and WrapupOutMode is the wrapup mode variable for outgoing calls. The valid values for these parameters are:

- REQUIRED
- OPTIONAL
- NOTALLOWED

The default value for both WrapupInMode and WrapupOutMode is REQUIRED if not included in the Cisco.ini file.

When the wrapup mode is set to REQUIRED, for either incoming or outgoing calls, the agent has no options but to go to the Wrapup state when the call ends. All agent state buttons are disabled. While in the wrapup state, the Ready and NotReady buttons should enabled for the agent to choose which state to go to once wrapup is complete.

When the wrapup mode is set to OPTIONAL, for either incoming or outgoing calls, the agent is able to enter any after call state—Wrapup, Ready or NotReady—by clicking the appropriate button.

When the wrapup mode is set to NOTALLOWED, for either incoming or outgoing calls, the agent is only able to enter the Ready or NotReady states. The wrapup button is disabled.

Nortel DMS-100

This switch is supported by CTI 4.1.9 and above.

Nortel Meridian

The Meridian does not support the NotReady state feature.

Rockwell Spectrum

- This switch is supported by CTI 4.1.6 release and above.
- In order for an Agent to successfully Login:
  - The peripheral type needs to be specified in the Cisco.ini file.

```
[Service]

PeripheralType      ="PTSpectrum"
```

– PositionID, which is a Logical Workstation Number, needs to be configured to keep a PositionID value for every agent. The ciscoENU.def file must have this PositionID on LoginData mapped as follows:

```
[CmdData:LogInData]

Param.AgentID       ="{@AgentId}"

Param.NumECCVars    ="0"

Param.Password      ="{@AgentPin}"

Param.PositionID    ="265"


[Command:Login]

CmdData             ="LogInData"

DeviceCommand       ="LogIn"

Hidden              ="TRUE"

Order               ="2"

Title               ="Log &In"
```

The PositionID can also be provided via the Siebel VB script on Login command.

- The Rockwell Spectrum does not support the Call Delivered event.

In order for a screenpop to function correctly, DeviceEvent needs to be customized. If Auto Answer is configured on the switch, then all DeviceEvent="EventAnswer" need to be changed to DeviceEvent="EventEstablished" throughout the ciscoENU.def file.

- MakeCall is only available for agents in the NotReady state.
- The agent is ready to answer calls when in the Ready/Available state.
- By clicking the TransferComplete button, the agent who originates a conference can leave the conference and allow other parties to continue talking.

Siemens Hicom

- Supports a private service SingleStepTransfer, but only on the US/Canadian Release 6.5 and greater, not the international version. The PIM does not support this private service, so BlindTransfer is currently disabled until future PIM changes.
- Unconnected calls can not be placed on hold, so the call must be established in order to use the Hold command. The Hold button is disabled if the call is not connected.
- Consultative calls can not be placed on hold. The Hold button is disabled for consultative calls.
- After making a consultative call and while initiating a transfer/conference call, use the RetrieveCall Device command (stays for Reconnect) to drop the consultative call instead of using the ReleaseCall command; otherwise, a second attempt to transfer/conference the call will fail and place the original call on hold. The ReleaseCall command is disabled in this case, while the RetreiveCall command is enabled.