Smartnonus

MANUAL TÉCNICO

NONUS 2011®

Rev. (IT - DCM - 108v0)

Índice

1) Características Técnicas

A leitora Smartnonus atende aos requisitos elétricos, mecânicos, protocolos de transmissão e interoperabilidade definidos pelos padrões:

- ISO/IEC 7816
- PC/SC versão 1.0
- EMV 4.0 Level 1.

ESPECIFICAÇÕES	
Modelo	Smartnonus
Dimensões	
Altura	15,55mm
Largura	70mm
Profundidade	51mm
Peso (Incluído Cabo)	50 g., 0,050 Kg
Cumprimento Cabo	100 cm (10%)
Alimentação	Bus Powered - USB - 5V
Consumo	Aprox. 60ma
Temperatura de Operação	0 a 55°
Temperatura de Armazenamento	0 a 70°

SMART CARD	
Comunicação	USB 2.0 - Full Speed - 12Mbps
Protocolos	ISO 7816 T=0 e T=1
Tipos de Cartão	Cartões tipo A (5V), Cartões tipo B (3V) e Cartões de Memória
Durabilidade Mecanismo	100.000 inserções
Detecção de Estado	Detecção Inserção / Remoção. Detecção automática tipo Smart Card
Proteções	Curto Circuito, Proteção Térmica, Proteção ESD
Interface Usuário	Led Bicolor (Verde/Vermelho). Presença Cartão. Cartão Operacional.

PADRÕES / COMPATIBILIDADE	
ISO 7816	Card interface electrical characteristics compliant. Parts 1, 2 e 3
PC/SC	Interoperability Specification for ICC´s and Personal Computer Systems Version 1.0
EMV (MasterCard, Visa)	EMV V4.0 Level 1 Compliant

2) Características Físicas e Elétricas

2.1) Inserção do Cartão

Com a leitora montada o cartão deve ser inserido com os contatos voltados para cima.
O cartão permanece todo o tempo acessível ao operador.

Figura 2.1 - Posição correta de inserção do cartão.

Indicação dos Leds

Verde		O leitor está ligado.
Vermelho		Cartão está conectado
		A luz vermelha piscante indica que o cartão esta sendo lido / escrito.

2.2) Descrição dos Contatos Elétricos

As propriedades elétricas e mecânicas da interface do leitor Smartnonus são compatíveis com a norma ISO/IEC 7816-2. Os contatos da interface são compatíveis e estão dispostos de acordo com norma ISO/IEC 7816-3. O leitor suporta cartões ISO 7816, classe A e classe B.

C1 (SC0_VCC)	VCC - Alimentação	C5 (SC0_GND)	GND - Ground
C2 (SC0_RST)	RST - Sinal de Reset	C6 (NC C6)	VPP (ISO 7816) - Não utilizado
C3 (SC0_CLK)	CLK - Sinal de Clock	C7 (SC0_IO)	IO - Entrada / Saída Dados
C4 (SC0_C4)	C4 RFU* ISO 7816	C8 (SC0_C8)	C8 RFU* ISO 7816

* RFU - Reserved for Future Use (Reservado para uso futuro de acordo com ISO/IEC 7816). Os contatos C4 (SC0_C4), e C8 (NC C6 e SC0_C8) não são utilizados e estão isolados eletricamente.

2.3) Propriedades Elétricas dos Contatos.

As propriedades elétricas são compatíveis com a seção 4 da Norma ISO/IEC 7816-3:1997.
A leitora Smartnonus não implementa nenhuma propriedade elétrica adicional ou não compatível aos requisitos definidos na seção 4 do padrão ISO/IEC 7816-3.

Abreviações:

VIH - Nível Alto de tensão de entrada
VIL - Nível Baixo de tensão de entrada
Vcc - Tensão de Alimentação
VPP - Pino de Programação / Voltagem de Programação
VOH - Nível Alto de tensão de saída
VOL - Nível Baixo de tensão de saída
tR - Tempo de subida entre 10% e 90% da amplitude do sinal
tF - Tempo de queda entre 90% e 10% da amplitude do sinal
LIH : Nível Alto de corrente de entrada.
LIL : Nível Baixo de corrente de entrada.
Icc - Corrente de Alimentação
IOH - Nível Alto de corrente de saída.
IOL - Nível Baixo de corrente de saída.
CIN - Capacitância de entrada

2.3.1) I/O - C7 (SC0_IO)

Este contato é usado como saída (output - modo de transmissão) para transmitir dados para o cartão, ou como entrada (input - modo de recepção) para receber dados do cartão. O contato não é setado em nível lógico alto a menos que o pino de VCC esteja alimentado e estável.

2.3.1.1) Modo de Transmissão
Quando em modo de transmissão, o terminal enviará dados ao cartão de acordo com as características mostradas na tabela abaixo:

Símbolo	Condições	Mínimo	Máximo	Unidade
VOH	0 < IOH < 20uA, Vcc = min.	0.8 x Vcc	Vcc	V
VOL	-0.5mA < IOL < 0, Vcc = min.	0	0.4	V
tR e tF	CIN(Icc) = 30 pF max.	-	0.8	us
Perturbação do Sinal	Low	-0.25	0.4	V
	High	0.8 x Vcc	Vcc + 0.25	V

2.3.1.2) Modo de Recepção
Quando em modo de recepção, o terminal interpreta os sinais do cartão, de acordo com as características abaixo:

Simbolo	Mínimo	Máximo	Unidade
VIH	0.6 x Vcc	Vcc	V
VIL	0	0.5	V
tR and tF	-	1.2	us

2.3.2) VPP - C6 (NC C6)

VPP está eletricamente isolado.
A resistência entre NC C6 e qualquer outro contato é maior ou igual a 10M (Mega ohms), para uma tensão aplicada de 5Vcc.

2.3.3) CLK (C3 - SC0_CLK)

As características do Clock geradas pelo pino CLK, são descritas na tabela abaixo:

Símbolo	Condições	Minimo	Máximo	Unidade
VOH	0 < IOH < 50uA, Vcc = min.	Vcc - 0.5	Vcc	V
VOL	-50uA < IOL < 0, Vcc = min.	0	0.4	V
tR e tF	CIN(Icc) = 30pF max.	-	8% do Período do Clock	
Perturbação do Sinal	Low	-0.25	0.4	V
	High	Vcc - 0.5	Vcc + 0.25	V

2.3.4) Reset (C2 SC0_RST)

O sinal de Reset gerado pelo leitor apresenta as características elétricas descritas na tabela abaixo:

Símbolo	Condições	Minimo	Máximo	Unidade
VOH	0 < IOH < 50uA, Vcc = min.	Vcc - 0.5	Vcc	V
VOL	- 50uA < IOL < 0, Vcc = min.	0	0.4	V
tR e t	CIN(Icc) = 30pF max.	-	0.8	us
Perturbação do Sinal	Low	-0.25	0.4	V
	High	Vcc - 0.5	Vcc + 0.25	V

2.3.5) Vcc (C1 - SC0_VCC)

O pino Vcc fornece uma tensão de 5VDC. Entrega uma saída estável de corrente de 55ma.
O pino Vcc está protegido contra transientes e surtos de tensão originados pela operação interna do circuito do leitor ou de origem externa.
O Vcc é capaz de fornecer mais do que 55ma se requisitado em operação.

3) Sinais Eletrônicos e Protocolos.

A conexão, ativação e desativação dos sinais elétricos dos contatos da leitora são realizados de acordo com o padrão ISO/IEC 7816-3.
Isto inclui os procedimentos de Cold e Warm Reset. O diálogo entre a leitora e o cartão é conduzido através das seguintes operações consecutivas:

a) Conexão e Ativação dos contatos pela leitora.
b) Reset do Cartão
c) ATR (Answer to Reset) pelo cartão.
d) Transação (troca de informações) entre o cartão e a leitora.
e) Desativação dos contatos pela leitora.

3.1) Conexão e Ativação dos Contatos

O circuito elétrico da leitora não é ativado até os contatos estarem apropriadamente conectados na interface, isto evita possíveis danos ao cartão.
A ativação dos contatos pela interface da leitora, consiste das seguintes operações consecutivas:

a) RST (Reset) em nível lógico baixo.
b) Vcc ativo.
c) I/O em modo de recepção.
d) CLK (Clock) é ativado e estabilizado.

3.2) Reset do Cartão

3.2.1) Cold Reset

O procedimento de ativação por Cold Reset consiste das seguintes operações consecutivas.
Esta representado na figura 3a, e as indicações de tempo(ta,tb e tc) estão descritas de acordo com a norma ISO/IEC 7816-3.

a) Reset em nível lógico baixo e I/O em modo de recepção de dados.
b) Vcc estabilizado
c) Uma vez que Vcc esta estabilizado, é aplicado um Clock estável em CLK num dado tempo ta. (ta <= 200/f)
d) O Reset (RST) é mantido em nível lógico baixo por um período definido por ta + tb. Sendo tb < 400 ciclos de clock (tb < 400/f).
e) O ATR (Answer to Reset) é monitorado na linha de I/O por um período (tc) definido por 400 e 40.000 ciclos de clock depois de tb
(400/f < tc < 40.000/f).

Sendo f = frequência do clock. (6.xx MHz)

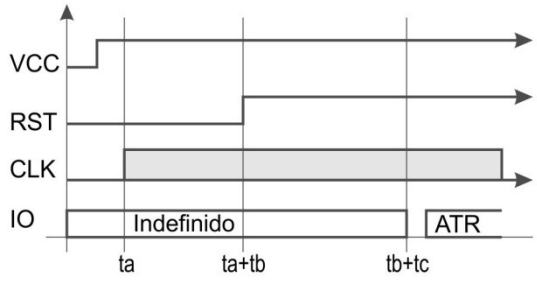


Figura 3a

3.2.2) Warm Reset

O procedimento de ativação por Warm Reset consiste das seguintes operações consecutivas.
Esta representado na figura 3b, e as indicações de tempo(td,te e tf) estão descritas de acordo com a norma ISO/IEC 7816-3.

a) Vcc ativo e estabilizado, Reset em nível lógico alto e Clock (CLK) ativo e estabilizado.
b) RST é dirigido para nível lógico baixo, a qualquer tempo T para iniciar um Warm Reset. O Reset é mantido em nível lógico baixo por 400 ciclos de clock até um tempo td. td = T + te. 400/f < te.
c) O terminal (leitora) mantém I/O em modo de recepção.
d) O terminal dirige o RST para nível lógico alto após 400 ciclos de clock (td).
e) O cartão responde com um ATR (Answer to Reset) dentro de 40.000 ciclos de clock (tf < 40.000/f).

Sendo f = frequência do clock. (6.xx MHz)

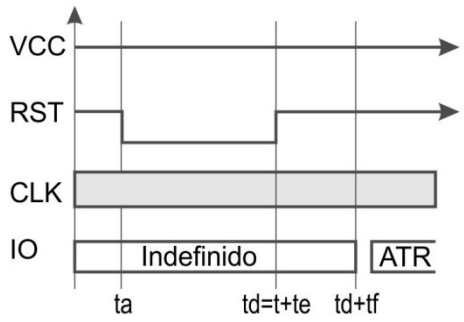


Figura 3b

3.3) Desativação dos Contatos

Quando a transação (troca de informações) é terminada ou abortada (cartão não respondendo ou cartão removido) os contatos são desativados através das seguintes operações:

a) RST em nível lógico baixo.
b) CLK em nível lógico baixo.
c) I/O em nível lógico baixo.
d) Vcc inativo.

3.4) ATR

Os dados entre o leitor e o cartão são trocados por uma única linha bidirecional (half-duplex).
A comunicação com o cartão é sempre iniciada pela leitora. O que significa que o cartão nunca envia dados sem um estimulo externo.
Em seguida da ativação elétrica o cartão executa um Reset e envia um ATR (Answer to Reset) a leitora.
O ATR é independente de protocolo de transmissão e permite a leitora avaliar o ATR que contém parâmetros relacionados ao cartão e a transmissão de dados.

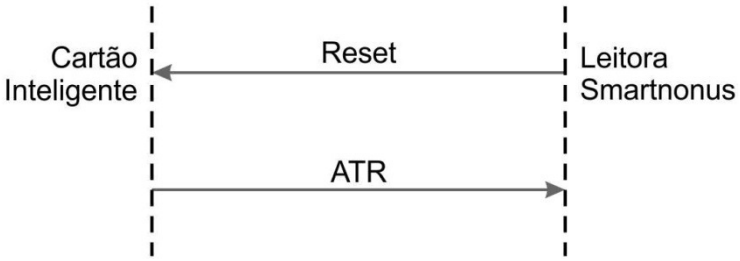


Figura 3.4a - Após o Reset, o cartão envia uma resposta ATR.

Uma operação de reset resulta em uma resposta ATR do cartão definida pela norma ISO/IEC 7816:

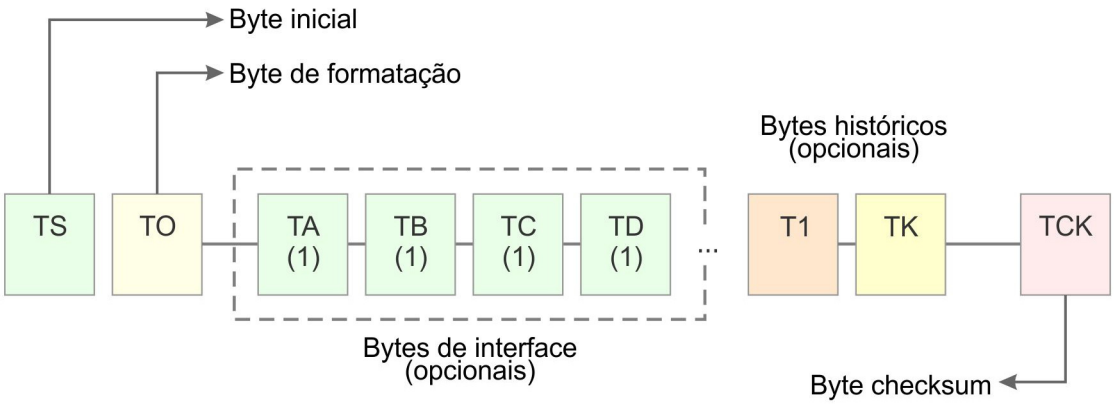


Figura 3.4b - Conteúdo do ATR.

Segue abaixo uma breve descrição dos bytes que compõe o ATR, para mais informações consulte a norma ISO/IEC 7816-3.

TS : Byte Inicial
O byte TS permite sincronização do frame serial e determina se o cartão trabalha com convenção direta (3B) ou inversa (3F)

T0 : Byte de formatação.
O byte T0 determina quantos bytes de interface e quantos bytes de histórico estão presentes no ATR
Os bits b8 a b5 determinam se os bytes TA, TB, TC e TD estarão presentes no ATR, por exemplo, se o bit b8 estiver setado, significa que o byte TD estará presente.

Os bits b4 a b1, indicam o número de bytes históricos presentes no ATR.

b8	b7	b6	b5	b4	b3	b2	b1	Descrição
				x	x	x	x	Numero de Bytes Históricos
			1					TA(1) presente
		1						TB(1) presente
	1							TC(1) presente
1								TD(1) presente

TA : Global, Interface Bytes, Códigos F e D
Os bits b8 a b5 (bits mais significativos), informam a taxa de divisão da frequência do Clock (F).
Os bits b4 a b1 (bits menos significativos), informam o fator de ajuste do baud rate (D).

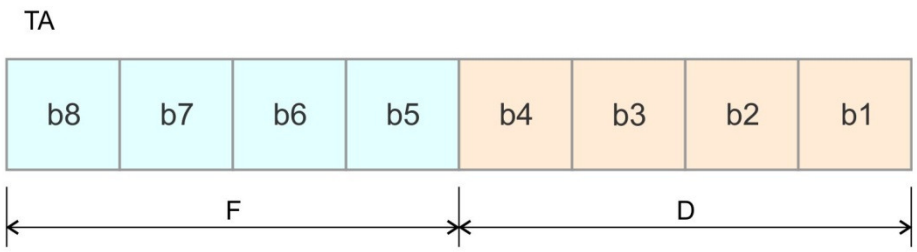


Figura 3.4c - Conteúdo do byte TA.

O Smartnonus suporta diferentes velocidades de baud rate na comunicação com o cartão, ex:.

9600 bps
19200 bps
38400 bps
57600 bps
115200 bps

Porém a seleção de baud rate não esta limitada a relação acima, para a leitora Smartnonus aceitar o parâmetro, as condições a) e b) abaixo devem ser verdadeiras.
A velocidade pode atingir no máximo 31 períodos de clock:

a) F/D deve ser inteiro, ou divisível.

Exemplo FI = 1 e DI = 8, F = 372 e D = 12, F/D = 31 - baud rate suportado.
Exemplo FI = 1 e DI = 4, F = 372 e D = 8, F/D = 46.5, - baud rate não suportado

b) F/D deve ser maior ou igual a 31 períodos de clock.

Estas velocidades e condições são verdadeiras para T=0 ou T=1. (Vide 3.6)

TB : Códigos P e I
Quando aplicável, P e I definem voltagem e corrente referentes ao pino de programação VPP (não utilizado).

TC : Código N
Guardtime extra solicitado pelo cartão.
Antes de receber o próximo caractere o cartão solicita um delay(atraso) de pelo menos (12+N) etu(elementar time unit) desde o inicio do caractere anterior.

TD : Bytes de Interface Adicionais, Protocolo
O primeiro nibble do byte TD, semelhante ao byte T0, a cada passagem, informa se haverão bytes de interface adicionais (TA2,TB2,TC2,TD2...TA3..TB3...e assim por diante).
Isto é útil quando o cartão possui mais de um protocolo ou opções disponíveis.
O segundo nibble do byte TD informa o primeiro protocolo disponível (T=0, T=1)

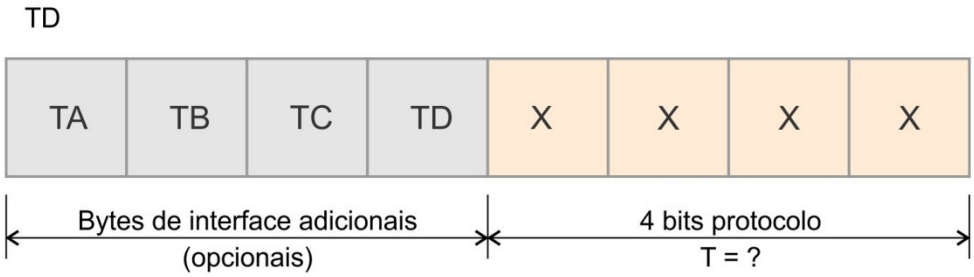


Figura 3.4d - Conteúdo do byte TD.

T1,T2...TK - Bytes de histórico
Estes bytes podem conter informações gerais, sobre os serviços ou características do cartão, por exemplo:
Fabricante, modelo de chip e S.O.
Dados Proprietários do emissor.
Dados adicionais (ex. Saldo)

TCK : Byte de Validação ou Checksum
A presença do byte de checagem TCK dependerá do protocolo selecionado.

3.5) PPS (Protocol and Parameter Selection)

Uma vez que um cartão pode especificar mais de um protocolo e parâmetros disponíveis, é preciso definir qual protocolo e que parâmetros a leitora deverá utilizar.
O protocolo e parâmetros precisam estar definidos antes da primeira troca de dados.
O mecanismo do comando PPS (antigamente chamado PTS) permite que a leitora negocie o protocolo e parâmetros com o cartão.
A leitora Smartnonus suporta o processo de PPS de acordo com a norma ISO/IEC 7816-3.
O comando PPS precisa ser executado uma única vez e imediatamente após o ATR ser recebido.
Transmissões repetidas de comandos PPS são proibidas pela norma ISO.

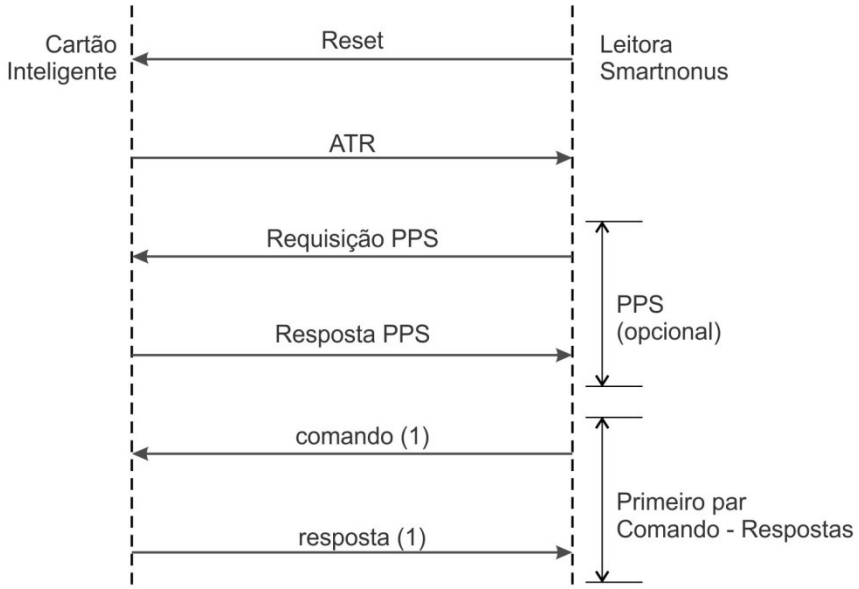


Figura 3.5a - Caso preciso, o comando PPS é executado logo após o ATR, antes do primeiro par comando-resposta.

Segue uma breve descrição da estrutura do comando PPS:

PPSS - Caractere Inicial
PPS0 - Caractere de Formato
PPS1, PPS2, PPS3 - Caracteres de Parâmetros
PCK - Caractere de validação ou checagem

PPSS : Caractere Inicial
O caractere PPSS informa o cartão que que a leitora esta iniciando uma requisição PPS.
O valor do caractere PPSS é 0xFF.

PPS0 : Caractere de formato
PPSO é um mapa de bits, e indica nos bits b5, b6 e b7 a presença dos caracteres de parâmetros PPS1, PPS2 e PPS3.
O bit b8 é reservado para uso futuro (RFU).
No nibble menos significativo(bits de 4 a 1) a leitora informa o protocolo desejado.

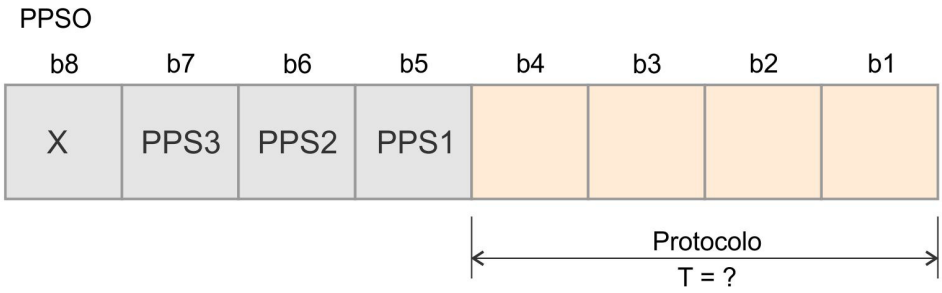


Figura 3.5b - Conteúdo do byte PPS0.

PPS1 : Parâmetros F e D
PPS1 codifica os parâmetros F e D do mesmo modo que o byte TA do ATR.
Se preciso a leitora indica a seleção de valores para F e D. Se PPS1 não for enviado, os valores F=1 e D=1 são assumidos.

PPS2 : Código N
PPS2 indica nos bytes b2 e b1 o suporte ao parâmetro N, Guardtime.
Se PPS2 for enviado e não ecoado na resposta, a leitora deverá rejeitar ou enviar um reset ao cartão.

O uso de PPS3 não esta definido, esta reservado para uso futuro (RFU).
PCK é o caractere de validação ou checagem.
Se o cartão receber e aceitar a requisição PPS como correta, deverá ecoar o comando enviado como confirmação.
Após uma troca bem sucedida de requisição e confirmação PPS, a transmissão utilizará o protocolo e parâmetros negociados.
Caso o cartão receba uma requisição PPS inválida, não enviará confirmação.
Se a leitora não receber confirmação, deverá rejeitar o cartão ou enviar um Reset.

3.6) Protocolos de Transmissão (T=0, T=1 e APDU)

Atualmente a leitora Smartnonus suporta os protocolos T=0 e T=1 definidos na norma ISO 7816-3:

* T = 0 (Transmissão half duplex, assíncrona, caracteres)
* T = 1 (Transmissão half duples, assíncrona, blocos)

O firmware da leitora não realiza a conversão do protocolo T=0 para T=1 ou T=1 para T=0.
Para T=0, o firmware lidará com a camada TPDU de T=0, o que significa que o firmware enviará o cabeçalho T=0 de 5 bytes e executará a ação de acordo com o byte de reconhecimento(ACK), e bytes de status (SW1, SW2). O tratamento do comando GET_RESPONSE não é realizado pelo firmware da leitora.
Do mesmo modo para T=1, o protocolo T=1 deve ser tratado pelo driver da aplicação (host), o firmware da leitora é responsável apenas por enviar ou receber dados do cartão inteligente.

3.6.1) Protocolo T = 0

No protocolo T=0 a leitora sempre inicia a transmissão dos dados.
O comando inicial enviado pela leitora ao cartão consiste em um cabeçalho de 5 caracteres.
Se o comando é recebido corretamente, o cartão envia um byte de reconhecimento(ACK).
Os dados podem ser transmitidos apenas em uma direção de cada vez, a direção da transmissão dos dados esta implícita na definição do comando, portanto tanto a leitora quanto o cartão conhecem a sequência antecipadamente.

Cabeçalho da Instrução para T=0

* CLA - a classe do comando ou instrução (o valor 0xFF é reservado para identificar uma sequência PPS)
* INS - o código da instrução (por exemplo "read memory")
* P1 - parâmetro 1 da instrução (por exemplo "endereço de memória")
* P2 - parâmetro 2 da instrução.
* P3 - Número de bytes do campo do bloco de dados da instrução.

A figura 3.6.1 abaixo exemplifica o funcionamento do protoloco T=0, no caso mais extenso, com envio de dados para processamento e dados gerados pelo cartão, para este caso faz-se necessário o uso do comando "GET_RESPONSE".
Se um comando for concluído e o cartão gerar apenas um código de retorno, sem um bloco de dados, a porção de "get response" na figura abaixo não ocorre.

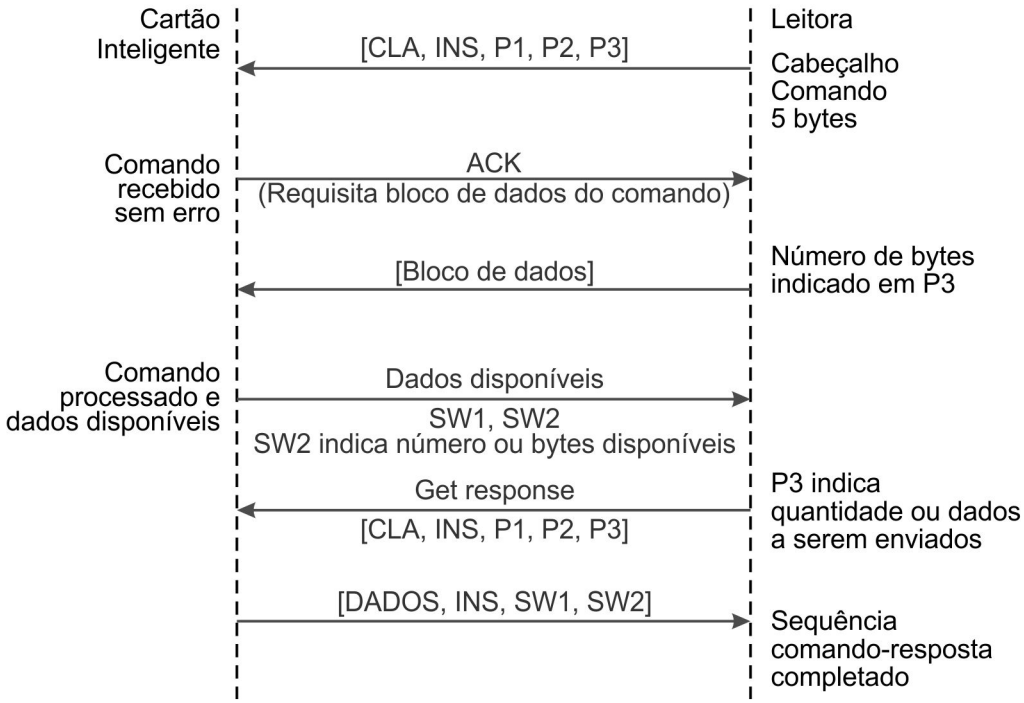


Figura 3.6.1 - Caso mais extenso de instrução no protocolo T=0.

Normalmente o valor do caractere ACK, é um eco do valor do byte de instrução (INS).

Visto que o protocolo T = 0 é orientado a caracteres (bytes), se for detectado um erro, a retransmissão do caractere incorreto deve ser feita imediatamente.
A detecção de erros no protocolo T = 0 é baseada exclusivamente no bit de paridade anexado a cada byte.

Existem 2 bytes de status SW1 e SW2. Estes bytes são enviados pelo cartão à leitora na conclusão do comando e indicam o estado corrente do cartão.
A resposta normal é:

SW1, SW2 = 0x90, 0x00

Quando SW1 = 0x6X ou 0x9X várias condições de erro podem ser reportadas pelo cartão.
A ISO 7816-3 define 5 condições de erro:

Valor de SW1	Descrição
0x6E	O cartão não suporta a classe de instrução.
0x6D	Byte de instrução (INS) inválido.
0x6B	Referência Incorreta
0x67	Cumprimento incorreto
0x6F	Sem diagnóstico em particular

3.6.2) Protocolo T = 1

A leitora Smartnonus suporta o protocolo T=1 de acordo com a norma ISO 7816-3.
O protocolo T=1, é um protocolo assíncrono, half duplex, orientado a blocos.
O protocolo T=1 esta localizado na camada 2 do modelo OSI (Data Link Layer).

O protocolo T=1 permite controle de fluxo, encadeamento de blocos, ressincronização e comunicações de tamanho ilimitado.
Possui um sistema de detecção e correção de erros mais sofisticado do que T=0.
Embora tenha vantagens em relação ao protocolo T=0, o seu uso deve ser ponderado pois ele exige um desenvolvimento mais complexo de software, e em alguns casos exigirá maior consumo de memória ROM e memória RAM, aumentando os custos do cartão inteligente.

A estrutura do bloco de comando no protocolo T=1 é composta por:

- Campo de Prólogo (Prologue Field)
- Campo de Informação (Information Field - APDU/Dados)
- Campo de Epilogo (Epilogue Field)

Campo de prólogo			Campo de informação	Campo de epílogo
Endereço do nó (NAD)	Byte de controle protocolo (PCB)	Tamanho (LEN)	Opcional (INF)	Código de detecção de erro (EDC)
1 byte	1 byte	1 byte	0-254 bytes	1 ou 2 bytes (LRC ou CRC)

Figura 3.6.2 - estrutura do bloco no protocolo T=1.

O byte PCB no prólogo define 3 tipos possíveis de blocos:

- Information Block (I-Block)
- Receive Ready Block (R-Block)
- Supervisor Block (S-Block)

Information Block (I-Block): Estrutura utilizada para transmitir comandos da aplicação entre o cartão e a leitora. Pode ainda atuar como um byte de reconhecimento (ACK) para o modo não encadeado.

Receive Ready Block(R-Block): O Receive Ready Block é usado como byte de reconhecimento (ACK/NAK) quando o protocolo esta enviando dados como uma sequência de blocos encadeados.

Supervisor Block(S-Block): É utilizado para estabelecer parâmetros de controle e efetuar ressincronização como resultado de alguma condição de erro.

O byte LEN, ainda no prólogo, define o número de bytes(se houver algum) no Campo de Informação do bloco. É permitida a faixa de valores de 0x00 a 0xFE, tamanho máximo do campo de 254 bytes.

O Campo de Informação (Information Field) é utilizado para transportar comandos ou dados da aplicação.

O Campo de Epílogo, contém o bloco de detecção de erros que pode ser tanto um LRC ou um CRC. O LRC utiliza 1 byte, enquanto o CRC utiliza 2 bytes. O código de detecção de erro é definido por caracteres de interface específicos.
Conforme discutido anteriormente (vide 3.4 ATR) os caracteres de interface são providos pelo ATR.
O protocolo T=1 utiliza 3 caracteres para estabelecer seus parâmetros antes de iniciar a comunicação.

Estes 3 bytes são (para i > 2):

* TAi = IFSC (IIFSC = Information Field Size for the Card, padrão = 32)
* TBi
  (bits 4 - 1) = CWI (padrão = 13)
  (bits 8 - 5) = BWI (padrão = 4)
* TCi
  (bit 1 = 1) = Código de Erro = CRC
  (bit 1 = 0) = Código de Erro = LRC, padrão

Os caracteres CWI e BWI estão relacionados a temporização entre caracteres e blocos.
A temporização entre caracteres permite a detecção de erros no tamanho de um bloco transmitido, enquanto a temporização entre blocos permite a detecção de um cartão sem resposta ou que ficou inoperante.

3.6.3) APDU

O APDU (Aplication Protocol Data Unit) é um protocolo de comando entre o cartão e a aplicação de acordo com a norma ISO 7816-4.
A implementação na leitora Nonus está em conformidade com a norma ISO 7816-4.
Mensagens APDU, compreendem 2 estruturas, uma é utilizada pela aplicação para enviar comandos para o cartão, e outra usada pelo cartão para enviar respostas de volta a aplicação.
A primeira é conhecida como APDU de comando (C-APDU) e a ultima como APDU de resposta (R-APDU). Para todo APDU de comando existe uma APDU de resposta correspondente.

Estrutura de um comando APDU:

Cabeçalho				Corpo opcional		
CLA	INS	P1	P2	LC	Dados	LE

Figura 3.6.3a - Estrutura de um comando APDU.

No cabeçalho do APDU encontramos:

* CLA - Classe da Instrução
* INS - Código da Instrução
* P1 e P2 - Parâmetros

Após o cabeçalho, existe um corpo opcional:

* LC - Dados enviados como complemento do comando para o cartão executar uma instrução especifica conforme definido no cabeçalho.
* DADOS - Dados enviados para o cartão. Estes dados podem ser estruturas TLV (Vide Apêndice A), suportada pela leitora Smartnonus.
* LE - Especifica o número de dados esperados pela aplicação na resposta do cartão.

Estrutura de uma resposta APDU.

Dados resposta	SW1	SW2

Figura 3.6.3b - Estrutura de uma resposta APDU.

O campo DADOS ou RESPOSTA é opcional.
Os bytes SW1 e SW2 são bytes de status de resposta e do cartão.

Abaixo os 4 diferentes casos possíveis de comando APDU

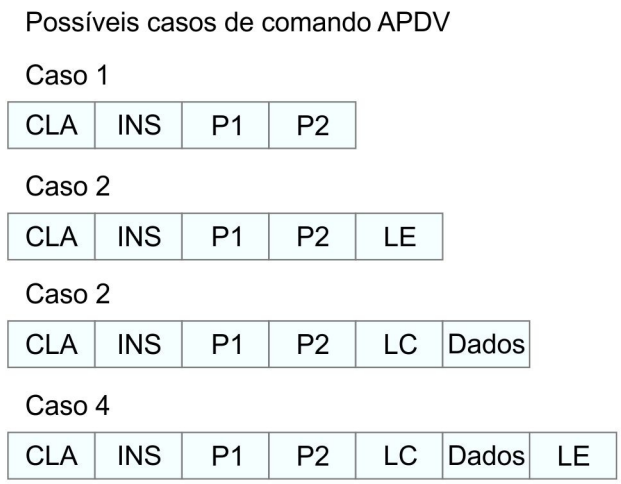


Figura 3.6.3c - Possíveis casos de comando APDU (T=0 ou T=1).

Abaixo os 2 casos possíveis de resposta APDU:

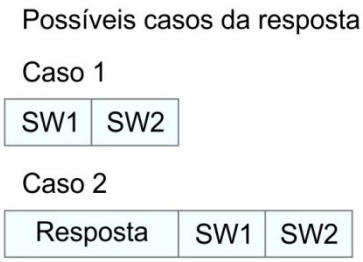


Figura 3.6.3d - Possíveis casos da resposta APDU (T=0 ou T=1).

A leitora Smartnonus é compatível com as demais normas de comunicação com os cartões conforme definido na ISO 7816-4.

O conceito de Sessão:
Para que o cartão mantenha dados de sessão ou dados dinâmicos, uma aplicação precisa estar selecionada.
Depois de um Reset, de um Power Down ou da seleção de uma nova aplicação, o conteúdo da memória RAM é descartado.
Todos os dados persistentes precisam ser explicitamente gravados em memória não volátil (EEPROM) do cartão.

Canais seguros (Secure Messaging):
A ISO 7816 define uma maneira de enviar (e/ou receber) comandos APDU com Secure Messaging.
Altera-se um bit do campo classe (CLA) da instrução e, ao invés do bloco de texto pleno, o cartão passa a esperar e/ou a devolver comandos com Secure Messaging.

4) Interoperabilidade com Computadores Pessoais

A leitora Smartnonus é compatível com a norma PC/SC 1.0 de Dezembro de 1997.
A norma PC/SC (Personal Computer / Smart Card) é um esforço para gerar uma especificação internacional para conectar cartões em PC´s. As empresas Bull, Hewlett-Packard, Microsoft, Schlumberger, Siemens Nixdorf, Gemplus, IBM, Sun, Verifone e Toshiba participaram da especificação desta norma.
Conforme especificado para as Leitoras(IFD - Interface Device), a Smartnonus adere a parte 3 ("Requirements for PC-Connected Interface Devices") e parte 4 ("Design Considerations and Reference Design Information") da especificação.
A especificação PC/SC é independente de plataforma, havendo implementações para os sistemas mais populares tais como o Microsoft Windows e diferentes distribuições do Linux, atendendo assim a grande maioria dos computadores pessoais.
A especificação PC/SC permite aos cartões inteligentes serem integrados a qualquer aplicação desejada, independente da linguagem de programação, desde que é suportada largamente por linguagens tais como C, C++, C#, Java e outros.

A Nonus disponibiliza através de seu site, ou em mídia quando requisitado, o Módulo de Interface na forma de "device drivers", implementando o Módulo da Interface e o Driver para o canal (I/O) USB, de forma que é possível aos Provedores de Serviço se comunicarem com cartão por meio da Leitora Smartnonus.
Os drivers da Smartnonus provém mecanismos de tratamento de erros, existem 2 partes de tratamento de erros, no canal USB e no protocolo do cartão inteligente. Estas tratativas são detalhadas nas seções a frente.
Os device drivers estão disponíveis para as principais versões de sistemas operacionais para PC´s, entre eles diversas versões do Windows, Linux e também para computadores MacOSx.

A figura abaixo ilustra a arquitetura PC/SC, a área pontilhada em vermelho representa o sistema de componentes da leitora.

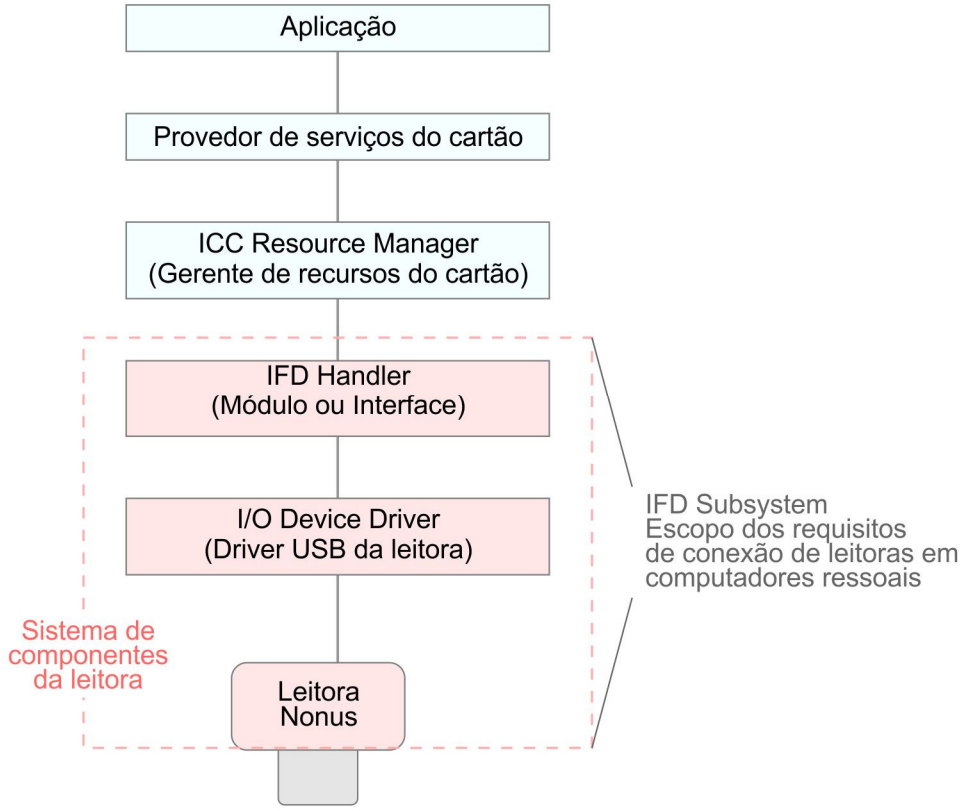


Figura 4a – Arquitetura PC/SC

4.1) Características Operacionais

O Módulo de Interface da Leitora Smartnonus, de acordo com a especificação PC/SC, apresenta um conjunto uniforme de serviços ao Provedor de Serviços do Cartão.
Suporta no mínimo, uma conexão lógica, ativa, entre uma aplicação e a leitora.
Esta funcionalidade não impede o gerenciamento de sessões, conforme definido no padrão ISO/IEC 7816, como os mecanismos de canais (channel mechanism) definidos na ISO 7816-4 ou a funcionalidade de endereços por nós (node addressing), associado ao protocolo T=1. Entretanto a implementação destas características relacionadas ao gerenciamento de sessões é responsabilidade do Cartão e Provedor de Serviços associado.
O Módulo de Interface da Leitora Smartnonus suporta múltiplas leitoras, e apresenta uma conexão lógica independente para cada leitora para requisições de um Provedor de Serviços de um Cartão. Também possibilita determinar a associação entre uma leitora específica e sua respectiva conexão lógica.

4.2) Enumeração das funcionalidades operacionais da Leitora

Através do Módulo de Interface a leitora Smartnonus provê uma interface ou serviço que permite enumeração das suas funcionalidades, mandatórias e opcionais.
Estas funcionalidades podem ser consultadas a qualquer instante. A informação retornada utilizando uma estrutura TLV (tag-length-value, vide apêndice A).
Abaixo, estão alistadas as informações que serão retornadas pela leitora Smartnonus.
Em adicional veja o apêndice C, Demonstração API Smartnonus, na sessão C.1.3, software demonstração Nonus que permite executar este serviço.

Classe de Informações	Elemento de Dado	TAG	Tam. Máximo	Dado Codificado
Fornecedor				
	Nome do Fornecedor	0x0100	32 bytes	“CASTLES”
	Especificação do Fornecedor para a Leitora (IFD)	0x0101	32 bytes	“EZ100PU”
	Especificação do Fornecedor para a versão da Leitora (IFD)	0x0102	4 bytes	DWORD codificado como 0X*MMmmbbbb* onde: *MM* = 00 *mm* = 00 *bbbb* = 0000
	Número de Série da Leitora (IFD)	0x0103	32 bytes	“”
Comunicação				
	ID do Canal	0x0110	4 bytes	0x00200000 DWORD codificado como 0x*DDDDCCCC* onde: *DDDD* = tipo do canal de dados *CCCC* = número do canal Os seguintes códigos são definidos para DDDD: 0x01 Porta Serial; *CCCC* é o número da porta 0x02 Porta Paralela; *CCCC* é o número da porta 0x04 Porta do Teclado PS/2; *CCCC* é igual a zero. 0x08 Canal SCSI; *CCCC* é o ID SCSI. 0x10 IDE; *CCCC* é o número do dispositivo. 0x20 USB; *CCCC* é o número do dispositivo. 0xF*y* Definição do Fornecedor, para y no intervalo de 0 a 15; *CCCC* é definido pelo fornecedor.

Classe de Informações	Elemento de Dado	TAG	Tam. Máximo	Dado Codificado
Protocolo (Veja parte 2 da especificação PC/SC)				
	Tipos de Protocolos Assíncronos suportados	0x0120	4 bytes	0x00000003 DWORD codificado como 0x*0RRRPPPP* onde: RRR é Reservado para Uso Futuro (RFU) e deve ser 0x000. PPPP codifica os tipos de protocolos suportados. Um '1' dado numa determinada posição de bit, indica suporte para o protoloco ISO. Exemplo: 0x00000003 indica suporte para T=0 e T=1. Este é o único valor em conformidade que deve ser retornado pelos dispositivos neste momento. Todos os outros valores (T=2, T=14, T=15, e assim por diante, estão fora desta especificação e devem ser tratadas pelos drivers supridos pelo fornecedor)
	CLK padrão	0x0121	4 bytes	3571 Padrão de CLK do cartão (ICC) em KHz codificados como valores inteiros. Por exemplo: 3.58 MHz é codificado como o valor inteiro 3580.
	CLK máximo	0x0122	4 bytes	3571 Frequência de CLK máxima suportada pelo cartão, codificada como valores inteiros.
	Taxa de transmissão de dados padrão	0x0123	4 bytes	9600 Taxa de transmissão de dados padrão em bps codificada como valores inteiros.
	Taxa de transmissão de dados máxima	0x0124	4 bytes	9600 Taxa de dados máxima suportada pelo canal I/O do Cartão
	IFSD máximo	0x0125	4 bytes	0xFC DWORD indicando o IFSD máximo suportado pela Leitora (IFD). Pelo menos 32,254 é recomendado.
	Tipos de Protocolos Síncronos suportados.	0x0126	4 bytes	N/A DWORD codificado como 0x*4RRRPPPP* onde: RRR is Reservado para Uso Futuro (RFU) e deve ser 0x000. PPPP codifica os tipos de protocolos suportados. Um '1' dados numa determinada posição, indica suporte para o protocolo associado. 0x0001 indica suporte para o protocolo "2-wire". 0x0002 indica suporte para o protocolo "3-wire". 0x0004 indica suporte para o protocolo "I²C" Todos os outros valores estão fora desta especificação e devem ser tratados pelos drives do

Classe de Informações	Elemento de Dado	TAG	Tam. Máximo	Dado Codificado
				fornecedor.
Gerenciamento de Energia				
	Gerenciamento de Energia suportado	0x0131	4 bytes	0 Se 0, o dispositivo não suporta "power down" enquanto o cartão esta inserido. Se diferente de zero a leitora tem suporte.
Características de Garantia de Segurança				
	Dispositivos de autenticação do usuário para o cartão.	0x0140	4 bytes	0x00000000 DWORD que é o resultado de uma operação "bitwise OR" executada nos seguintes valores: 0x00000000 Nenhum dispositivo 0x00000001 Reservado para uso Futuro (RFU) 0x00000002 Teclado Numérico (Pin Pad) 0x00000004 Teclado 0x00000008 Scanner de impressão digital. 0x00000010 Scanner de retina 0x00000020 Scanner de imagem 0x00000040 Scanner de padrão de voz 0x00000080 Dispositivo de display 0x0000dd00 dd é um dispositivo definido pelo fornecedor
	Dispositivo de Entrada de identificação do Usuário.	0x0142	4 bytes	0x00000000 DWORD que é o resultado de uma operação "bitwise OR" executado nos seguintes valores: 0x00000000 Nenhum dispositivo 0x00000001 Reservado para uso Futuro (RFU) 0x00000002 Teclado Numérico (Pin Pad) 0x00000004 Teclado 0x00000008 Scanner de impressão digital. 0x00000010 Scanner de retina 0x00000020 Scanner de imagem 0x00000040 Scanner de padrão de voz 0x00000080 Dispositivo de display 0x0000dd00 dd no intervalo de 0x01 to 0x40 selecionado pelo fornecedor para os dispositivos definidos pelo fornecedor. 0x00008000 Utilizado para indicar que entrada encriptada é suportada
Características Mecânicas				
	Características Mecânicas suportadas	0x0150	4 bytes	0x00000000 DWORD que é o resultado de uma operação "bitwise OR" executada nos seguintes valores:

Classe de Informações	Elemento de Dado	TAG	Tam. Máximo	Dado Codificado
				0x00000000 Nenhuma característica especial 0x00000001 Mecanismo permite "engolir" cartão 0x00000002 Mecanismo permite ejetar o cartão 0x00000004 Mecanismo permite capturar o cartão Todos os outros valores são Reservados para Uso *Futuro (RFU)*
Características definidas pelo Fornecedor		Devem ser usados valores no intervalo 0x0180–0x01F0	--	N/A

4.3) Eventos do Cartão

A leitora Smartnonus é capaz de identificar e notificar através do módulo de interface:

- Inserção do cartão inteligente.
- Remoção do cartão inteligente

Não há outros eventos de cartão que podem ser detectados, visto a leitora não suportar mecanismos adicionais, para reter, "engolir" ou confiscar os cartões.
O cartão permanece acessível ao usuário para remoção a qualquer momento.

4.4) Gerenciamento da Interface do Cartão Inteligente

O sistema de componentes da leitora é capaz de gerenciar a interface elétrica do cartão de acordo com a parte 2 da especificação PC/SC.
Isto inclui:
- Provisão de todos os sinais elétricos/lógicos para os contatos do cartão.
- Sequência de ativação e desativação (Reset, Cold and Warm Reset).
- Recuperação e Interpretação correta do ATR.

O sistema de componentes da leitora pode notificar o Gerenciador de Recursos do cartão se um ATR inválido foi transmitido.
Quando um ATR válido é transmitido e interpretado corretamente o sistema de componentes da leitora pode negociar os parâmetros de comunicação compatíveis entre a leitora e o cartão.
O módulo da interface provê uma interface ou serviço que torna disponível determinar o estado de um cartão a qualquer momento.

A informação retornada utiliza uma estrutura TLV (tag-length-value, vide apêndice A).
Em adicional veja o apêndice C, Demonstração API Smartnonus, na sessão C.1.3, software demonstração Nonus que permite executar este serviço.

Informação	TAG	Tamanho Máximo	Respostas (retornadas como inteiros)
Presença do Cartão (ICC)	0x0300	1 byte	0 = não presente 1 = cartão presente mas não "engolido" (aplicável apenas se a Leitora (IFD) suporta o recurso "engolir" o cartão) 2 = cartão presente (e "engolido" para Leitoras (IFD) que suportam o recurso "engolir" o cartão) 4 = cartão confiscado
Estado da Interface do Cartão (ICC)	0x0301	1 byte	BOOLEAN, 0 = contato inativo 1 = contato ativo
Sequência ATR	0x0303	32 bytes	Contém a Sequência ATR conforme retornada pelo Cartão (ICC)
Tipo do Cartão (ICC), baseado na sequência ATR	0x0304	1 byte	ISO/IEC 7816 ou desconhecido 0 = Tipo de Cartão desconhecido 1 = 7816 Assíncrono 2 = 7816 Síncrono Outros valores Reservados para Uso Futuro (RFU)

O sistema de componentes da leitora Smartnonus é responsável por determinar quando ocorrem erros irrecuperáveis de comunicação e notificar o Provedor de Serviços do cartão logicamente conectado.

O sistema de componentes da leitora pode distinguir entre 2 tipos de erros de comunicação:
- Cartão inoperante (não respondendo)
- Erro irrecuperável de comunicação.

Esta informação é tornada disponível ao Provedor de Serviços do cartão logicamente conectado.
Existem 2 partes de tratamento de erros, no canal USB e no protocolo do cartão inteligente.
Para o canal I/O USB, basicamente o tratamento de erros segue o padrão USB. Além do mais se os dados são para o cartão inteligente, existe um time-out. O time-out é diferente para os protocolos T=0 e T=1. O valor do time-out para T=0 e T=1 depende do padrão ISO 7816. Se os dados não são para o cartão inteligente existe um time-out fixado.
Para o protocolo do cartão inteligente, existem 2 partes, T=0 e T=1.
Para T=0, o mecanismo de tratamento de erros é baseado na checagem de paridade.
Para T=1, existem 3 mecanismos para tratamento de erros. O primeiro é o BWT time-out. O segundo é a perda de sincronização. O terceiro ocorre quando o cartão envia um Bloco-R para a leitora indicando um erro de transmissão (paridade incorreta ou erro de EDC) ocorre.

4.5) Suporte de Protocolo

De acordo com a definição de camadas da ISO 7816, o Sistema de Componentes da Leitora Smartnonus implementa as camadas física e de enlace de dados (ISO 7816 3 e 10).
A figura abaixo mostra o fluxo de informações entre o Provedor de Serviços do Cartão e o Sistema de Componentes da Leitora Smartnonus.
O Sistema de Componentes da Leitora encapsula ou esconde do Nível de Aplicação todos os detalhes do protocolo e apresenta uma interface padrão baseada na estrutura de comandos e respostas da norma ISO 7816-4.

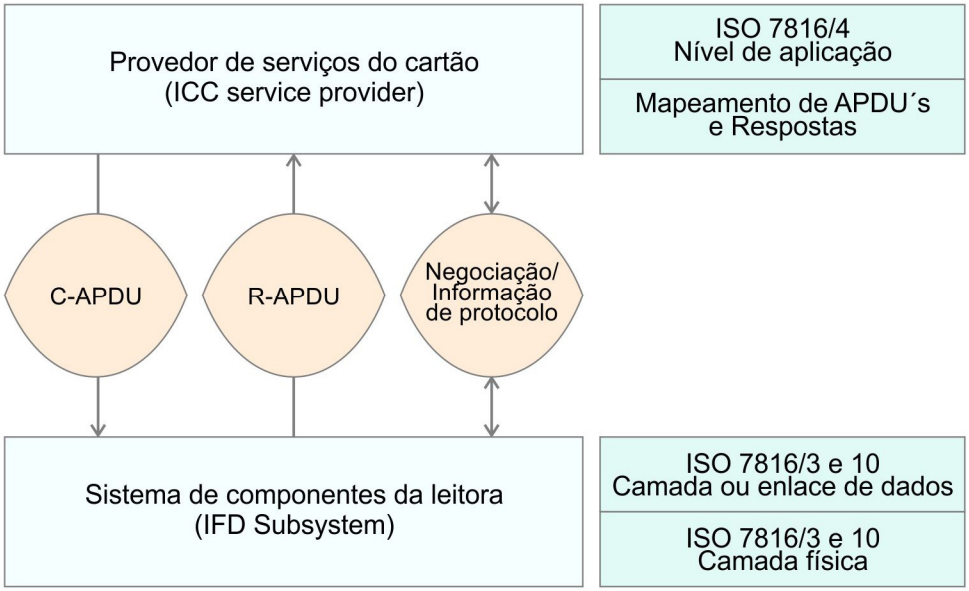


Figura 4.5a.

4.5.1) Negociação do Protocolo

A leitora Smartnonus suporta os protocolos T=0 e T=1 (Vide 3.6).
Suporta a frequência padrão dentro do intervalo de 1 a 5 MHz.
A leitora aguarda até que uma aplicação estabeleça uma conexão lógica antes de negociar os parâmetros do protocolo (PPS).
As requisições de conexão contém indicadores do protocolo desejado, bem como parâmetros que devem ser otimizados ou deixados no valor padrão.
O módulo da interface da leitora Smartnonus provê uma interface ou serviço que permite ao Provedor de Serviços do Cartão enumerar as configurações de protocolo e os parâmetros disponíveis. A informação retornada, codificada na tabela abaixo, utiliza uma estrutura TLV (tag-length-value, vide apêndice A).
Em adicional veja o apêndice C, Demonstração API Smartnonus, na sessão C.1.3, software demonstração Nonus que permite executar este serviço.

Elemento de Dado	TAG	Tam. Máximo	Somente Leitura	Comentários
Tipo de Protocolo Corrente	0x0201	4 bytes	X	DWORD codificado da mesma maneira que os Tipos de Protocolos Disponíveis. É ilegal especificar mais do que um protocolo neste valor.
CLK Corrente	0x0202	4 bytes	X	Frequência de CLK corrente do Cartão em KHz codificada como valores inteiros. Exemplo: 3.58 MHz é codificado como o inteiro 3580.
F Corrente(Fator de Conversão do Clock)	0x0203	4 bytes	X	F codificado como inteiro. (Pode ser modificado através de PPS)
D Corrente (Fator de Conversão do Bit Rate)	0x0204	4 bytes	X	D codificado como inteiro. (Pode ser modificado através de PPS)
N Corrente (Guard Time)	0x0205	4 bytes	X	N codificado como inteiro. (Pode ser modificado através de PPS)
W Corrente (Work Waiting Time)	0x0206	4 bytes	X	W codificado como inteiro. Válido apenas se protocolo corrente é T=0.
IFSC Corrente (Tamanho do campo de informação do cartão)	0x0207	4 bytes	X	IFSC codificado como inteiro. Válido apenas se protocolo corrente é T=1.
IFSD Corrente (Tamanho do campo de informação da Leitora)	0x0208	4 bytes		IFSD codificado como inteiro. Válido apenas se protocolo corrente é T=1.
BWT Corrente (Block Waiting Time)	0x0209	4 bytes	X	BWT codificado como inteiro. Válido apenas se protocolo corrente é T=1.
CWT Corrente (Character Waiting Time)	0x020A	4 bytes	X	CWT codificado como inteiro. Válido apenas se protocolo corrente é T=1.
Codificação de EBC corrente	0x020B	4 bytes	X	EBC codificado como: 0 = LRC 1 = CRC Válido apenas se protocolo corrente é T=1.

4.5.2) PC/SC : Suporte ao protocolo T=0

De acordo com a especificação PC/SC, a partir da perspectiva do Provedor de Serviços do Cartão, o SCL (Sistema de Componentes da Leitora) Smartnonus aceita os casos curtos (short cases) de APDU conforme definido na ISO 7816-4 e retorna assincronamente as respostas do cartão.

O SCL da Smartnonus é ainda responsável por detectar e responder erros de camadas físicas (erros de paridade de bits) conforme requerido pela norma ISO/IEC 7816.
Em termos de resposta de processamento do cartão o SCL precisa apenas processar um byte de ACK.

A leitora irá então enviar os dados restantes associados ao último cabeçalho de comando ou reter dados do cartão. Neste caso o sistema não irá retornar um byte de ACK para o Provedor de Serviços do Cartão.
Conforme a norma PC/SC, a leitora Smartnonus não interpreta ou processa outros bytes de resposta do cartão, visto que estes são geralmente específicos da aplicação e devem ser retornados para o Provedor de Serviços do Cartão.

4.5.3) PC/SC : Suporte ao protocolo T=1

De acordo com a especificação PC/SC, a partir da perspectiva do Provedor de Serviços do Cartão, o SCL (Sistema de Componentes da Leitora) Smartnonus deve aceitar APDU´s, construir os blocos T=1 necessários para transmitir estes APDU´s e retornar assincronamente as respostas do cartão.
O SCL registra, a cada transmissão, quem tem o direito de transmitir. Por exemplo, se o cartão tem o direito de transmitir, o SCL retorna um erro caso o Provedor de Serviços do Cartão tente iniciar uma transmissão em bloco.

O SCL é responsável por detectar os seguintes erros:
- Erro de transmissão (paridade incorreta ou erro de EDC) indicados pelo nibble menos significativo do PCB ou do R-Block enviado pelo cartão.
- BWT (Block Waiting Time) time out
- Perda de sincronização

Quando um erro é detectado, é interpretado pelo SCL que então retransmite automaticamente o último bloco enviado. No máximo são realizadas 3 tentativas. Após isto o cartão é desativado, e o provedor de serviços do cartão é informado de que um erro irrecuperável ocorreu.
Todos os outros erros são de responsabilidade do cartão ou do Provedor de Serviços do Cartão.
Abaixo segue o procedimento implementado pelo SCL Smartnonus de acordo com a especificação PC/SC:

1) Se não há resposta do cartão para um bloco enviado pelo SCL dentro do tempo definido para BWT (Block Waiting Time) o SCL segue de acordo com o cenário 33 ou 35 definido na norma ISO 7816-3 anexo A para tentar recuperar a comunicação. Se este recurso falhar, o SCL desativa o cartão e informa o Provedor de Serviços do cartão que um erro irrecuperável ocorreu.

2) Se um bloco inválido é recebido como resposta a um R-Block, o remetente deve retransmitir um R-Block.

3) Se um S-Block(response) não for recebido em resposta a um S-Block(request) o remetente deve retransmitir um S-Block(request).

4) Se um bloco inválido é recebido como resposta a um S-Block(response), o SCL transmite um R-Block com bit 5 = número sequencial do próximo I-Block esperado.

5) Se o SCL detectar uma condição de underrun ou overrun, aguarda o maior CWT ou BWT antes de transportar o ultimo bloco.

6) Quando um cartão envia um S-Block(IFS request) e recebe uma resposta inválida, retransmite o bloco uma única vez para obter um S-Block(IFS response) e em seguida permanece no modo de recepção.

A única ocasião em que o SCL pode iniciar um bloco é para estabelecer um valor para o parâmetro IFSD maior do que o default de 32 bytes. Neste caso o SCL realiza esta operação em seguida da negociação do protocolo T=1 e antes da transmissão do primeiro bloco do Provedor de Serviço do cartão.

O SCL Smartnonus provê suporte para funções encadeadas, conforme descrito na ISO 7816-3, o que permite a transmissão de informações maiores do que o parâmetro IFSC(IFSD).
O SCL também provê suporte para sessões lógicas usando o mecanismo de Node Address associado ao protocolo T=1.

4.6) Gerenciamento de Energia no Cartão

A leitora Smartnonus não possui funções de Gerenciamento de Energia.

4.7) Características Mecânicas

O mecanismo implementado pela leitora Smartnonus é um mecanismo de inserção manual.
A leitora Smartnonus não possui mecanismos de trava do cartão, não permite engolir, ejetar ou confiscar o cartão.
O cartão permanece todo o tempo acessível ao operador.

4.8) Dispositivos Adicionais de Segurança

A leitora Smartnonus não implementa, nem suporta expansão para dispositivos adicionais de segurança, tais como:

- Teclado numérico (Pin Pad)
- Teclado alfanumérico
- Dispositivo biométrico
- Tela ou Display

4.9) Características especificas do Fabricante

A leitora Smartnonus não implementa características, funcionalidades ou extensões proprietárias do fabricante.

4.10) Interface de Comunicação USB

A leitora Smartnonus utiliza o canal (I/O) USB (Universal Serial Bus), suportando comunicação de dados bidirecional. Através deste canal (I/O) a leitora implementa as funcionalidades necessárias para suportar o componentes do módulo de Interface (IFD Handler Interface)
A leitora Smartnonus é compatível com as especificações do padrão PC/SC Parte 3 e 4, não implementa o padrão USB CCID.
Na camada de função do protocolo USB (Function Layer) a leitora Smartnonus é implementada como um dispositivo periférico (ICC Reader Device), vide figura abaixo:

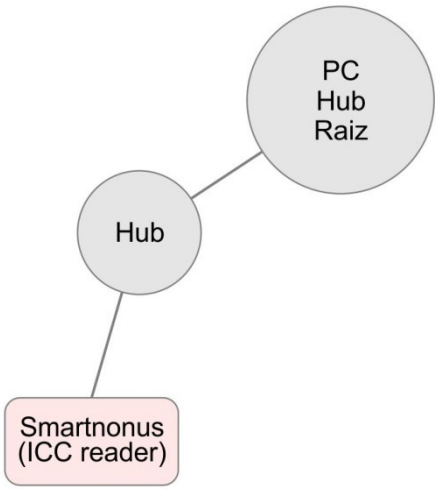


Figura 4.10a.

Abaixo estão alistadas as funcionalidades providas pela leitora Smartnonus como dispositivo ou periférico USB.

Função	Descrição
Ativação da leitora	Ativação automática quando a leitora é conectada (USB Hot-Plugging)
Detecção da inserção do cartão	Detecção da inserção ou remoção do cartão e geração de evento para o PC.
Gerenciamento dos contatos do cartão	Ativação automática dos contatos do cartão durante a inserção, desativação da leitora durante a remoção do cartão.
Inicialização do cartão	A leitora executa um Cold Reset na inserção e ativação de um cartão, executa a leitura da sequência ATR, quando presente, e envia o ATR ao PC. Se o tempo de resposta do ATR é excedido, a leitora informa o tipo de cartão como desconhecido.
Suporte de protocolo	A leitora implementa a camada física, incluindo a tratativa obrigatória de erros de paridade do protocolo T= 0.
Gerenciamento do canal I/O	Em seguida às sequências de Reset e ATR, a leitora permite ao PC o controle da transmissão. A leitora irá transmitir ao cartão, quando um bloco de dados for recebido do PC. Em seguida a leitora retorna os bytes recebidos do cartão para o PC até todo o bloco de dados ser recebido.

5) Manuais de Instalação

Seguem abaixo os manuais de instalação dos drivers da leitora Smartnonus para diversos Sistemas Operacionais.
Os drivers mais atuais podem ser obtidos no endereço: www.nonus.com.br/smartnonus
Cada Sistema Operacional abaixo apresenta a versão do driver disponível até a data de publicação desta versão do manual, bem como os requisitos de ambiente operacional necessários para compatibilidade.

5.1) Microsoft Windows 7 (32 e 64 bits)

Requisitos Mínimos de Ambiente Recomendados:
Microsoft Windows 7 (Sistema Operacional de 32 ou 64 bits)
Processador: 1GHz (ou superior)
Memória: Mínimo de 512MB
Espaço em Disco: 20MB
Versão do Driver: 3.1.7.0

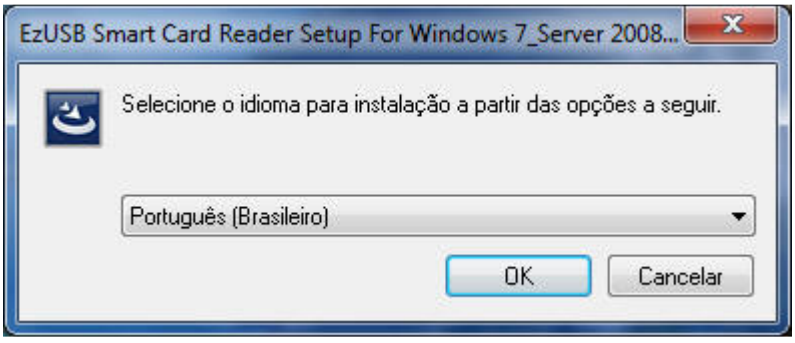
Mantenha a leitora Smartnonus **desconectada** durante a instalação dos drivers para o Microsoft Windows 7.
Você precisará de direitos administrativos para prosseguir com a instalação dos drivers para a leitora Smartnonus.

Os drivers mais atuais estão disponíveis para download na página da leitora Smartnonus na Internet em www.nonus.com.br/smartnonus.
Selecione o arquivo de acordo com a sua versão do Windows 7, entre 32 bits (setup7-32.exe) e 64 bits (setup7-64.exe).

Quando o seu navegador concluir o download, poderá executar diretamente o programa de instalação ou salvar o arquivo numa pasta local, como por exemplo sua pasta "Documentos".

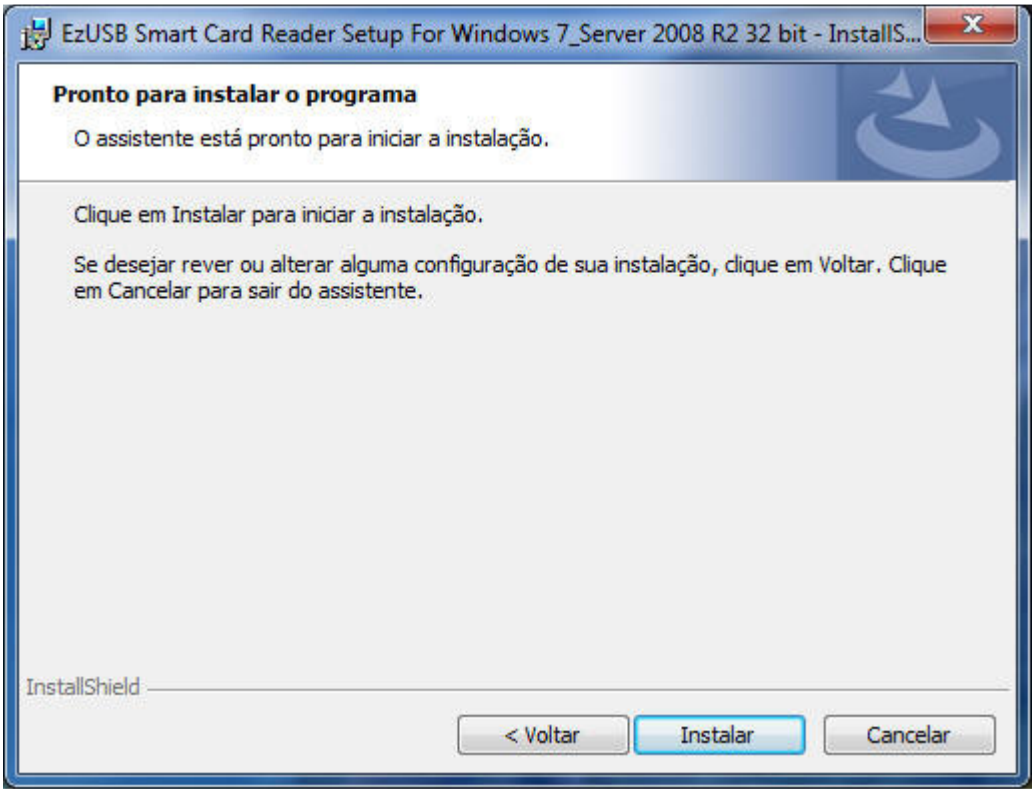
5.1.1) Execute o arquivo de instalação a partir de uma pasta local ou da página da leitora Smartnonus.

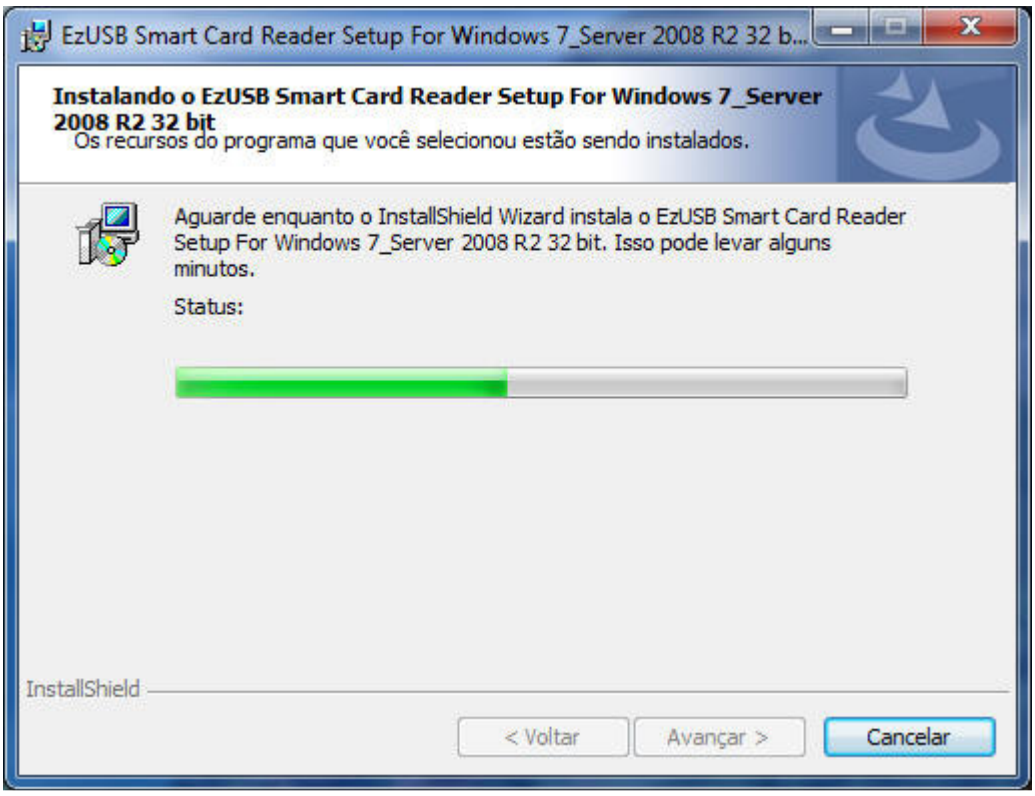
Selecione o idioma de instalação, escolha Português (Brasileiro).



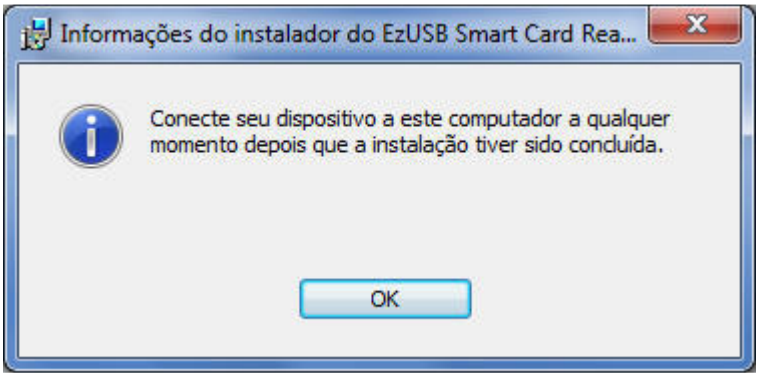
5.1.2) No formulário seguinte, aponte e clique no botão "Avançar".

5.1.3) No formulário seguinte, confirme a instalação clicando no botão "Instalar".

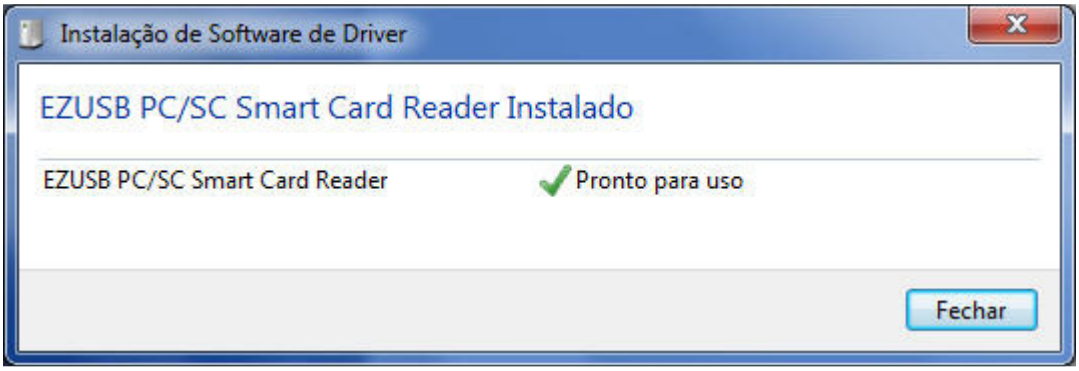




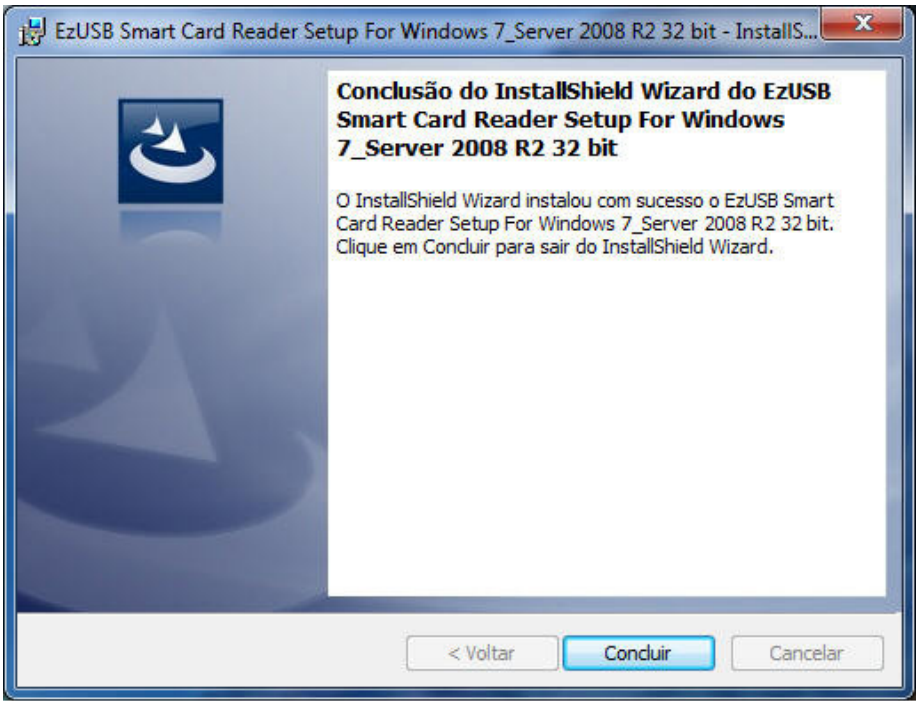
5.1.4) O programa de instalação copiará os arquivos do driver da leitora Smartnonus.
No final do processo de instalação, você receberá um aviso, neste instante **conecte** a leitora Smartnonus numa porta USB livre de seu computador, e clique no botão OK.



Quando conectar o leitor na porta USB, os drivers de instalação agora serão encontrados pelo Sistema Operacional.



5.1.5) Por fim clique em "Concluir".

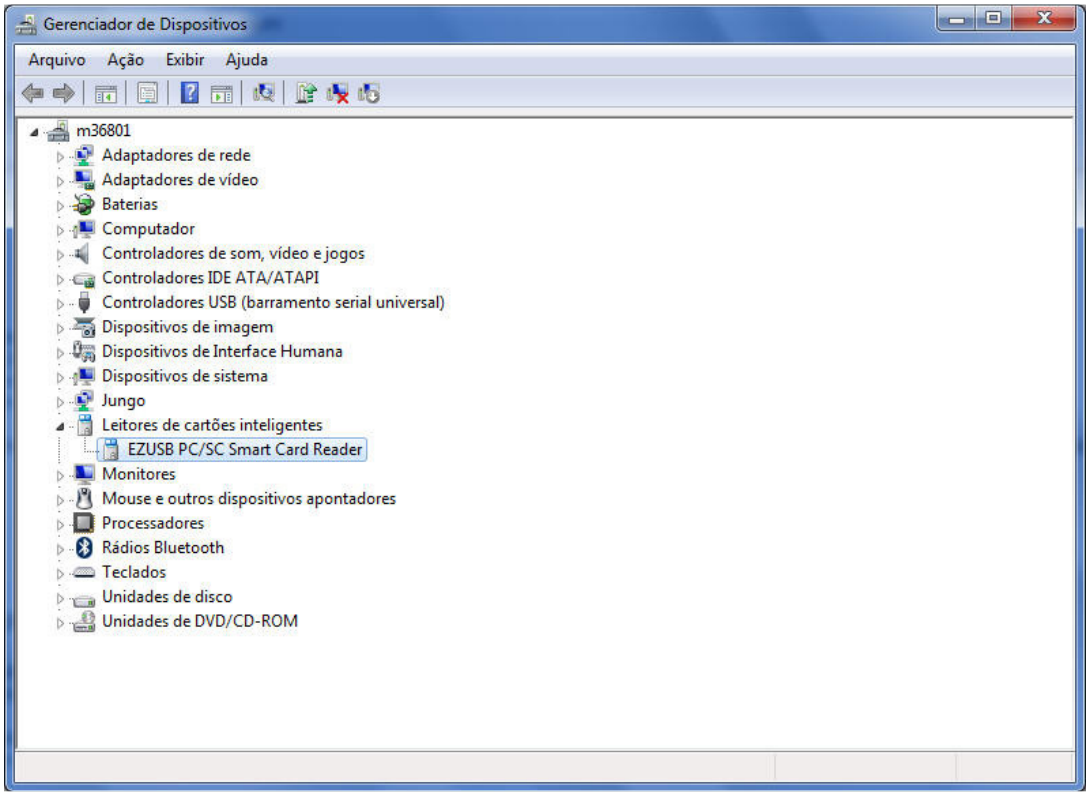


5.1.6) Poderá ainda fazer uma verificação adicional de instalação bem sucedida.

Clique no menu "Iniciar", e na caixa "Pesquisar programas e arquivos", digite: Gerenciador de Dispositivos.
Abra o programa localizado.

Ou se preferir, aponte e clique no Menu Iniciar, clique em Painel de Controle.
Em seguida clique em "Sistemas e Segurança".
Localize o ícone Sistema e clique em Gerenciador de Dispositivos.

Com o Gerenciador de Dispositivos aberto, clique para expandir o grupo Leitoras de Cartões Inteligentes.
Poderá visualizar a leitora Smartnonus descrita como EZUSB PC/SC Smart Card Reader.



5.2) Microsoft Windows Vista

Requisitos Mínimos de Ambiente Recomendados:
Microsoft Windows Vista (Service Pack 1)
Processador: 1GHz (ou superior)
Memória: Mínimo de 512MB
Espaço em Disco: 20MB
Versão do Driver: 3.1.7.0

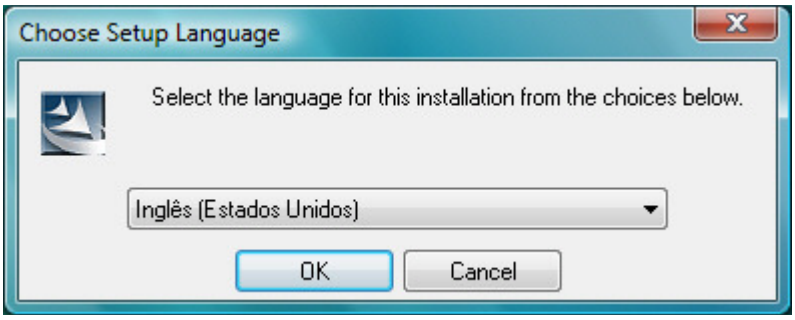
Mantenha a leitora Smartnonus **desconectada** durante a instalação dos drivers para o Microsoft Windows Vista.
Você precisará de direitos administrativos para prosseguir com a instalação dos drivers para a leitora Smartnonus.

Os drivers mais atuais estão disponíveis para download na página da leitora Smartnonus na Internet em www.nonus.com.br/smartnonus.
Selecione o arquivo para o Microsoft Windows Vista (setup-vista.exe).

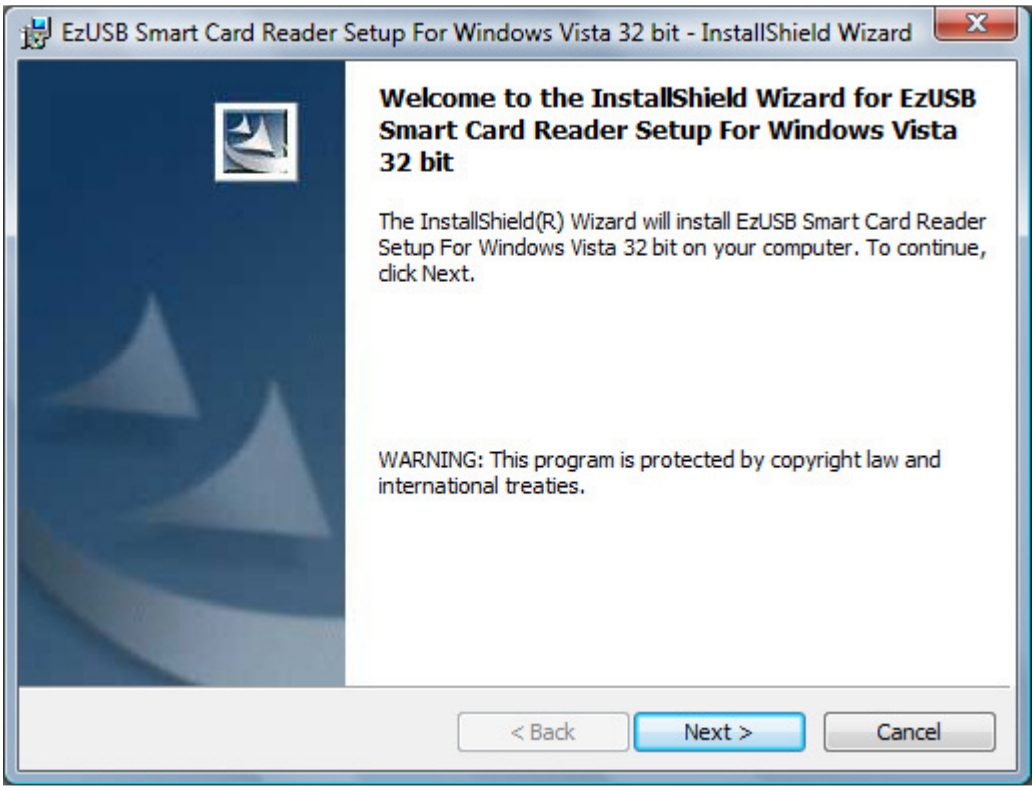
Quando o seu navegador concluir o download, poderá executar diretamente o programa de instalação ou salvar o arquivo numa pasta local, como por exemplo sua pasta "Documentos".

5.2.1) Execute o arquivo de instalação a partir de uma pasta local ou da página da leitora Smartnonus.

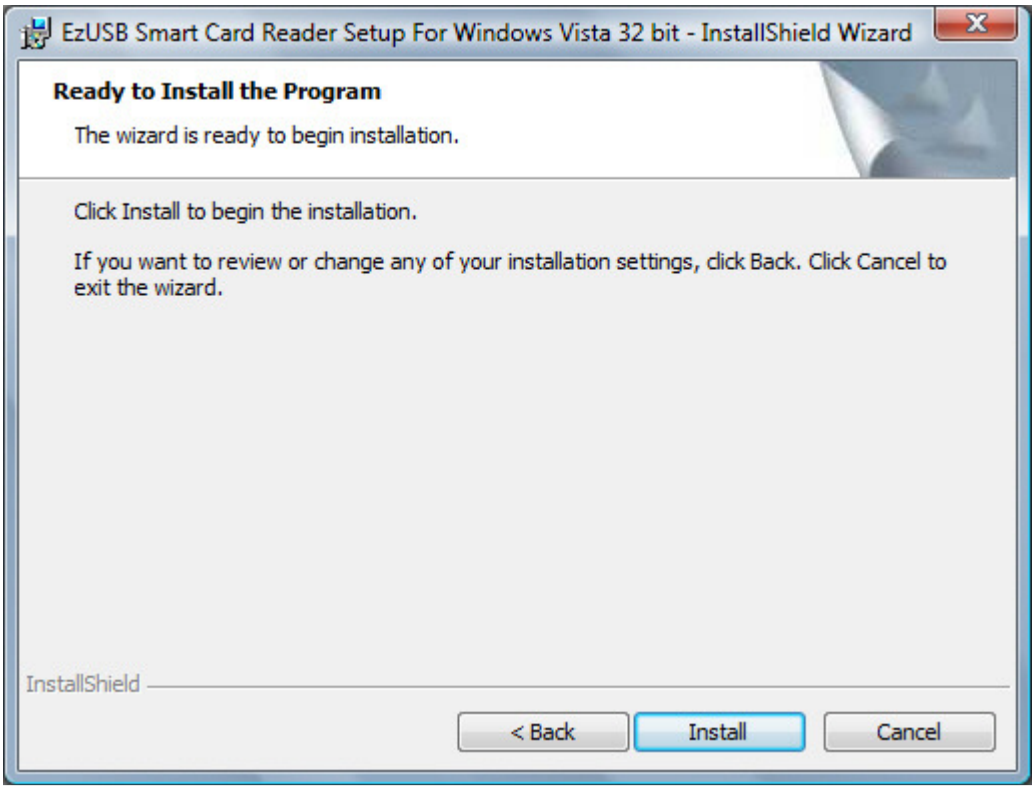
Selecione o idioma de instalação, escolha Português (Brasileiro), se estiver disponível, ou Inglês (Estados Unidos).

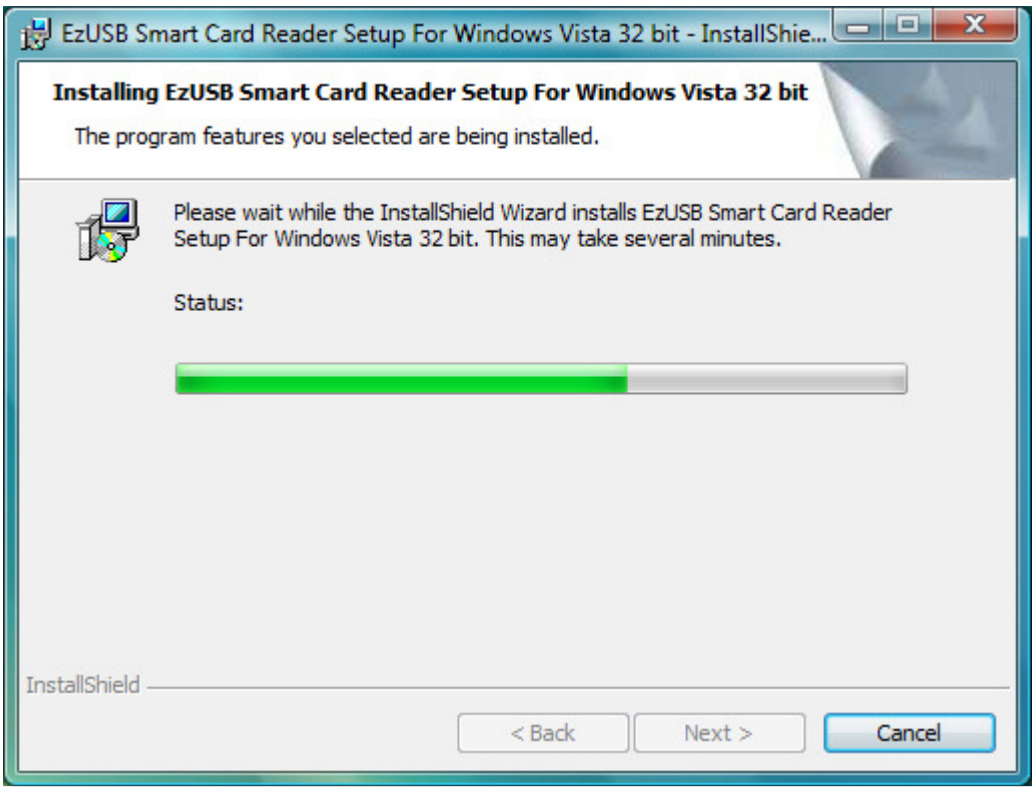


5.2.2) No formulário seguinte, aponte e clique no botão "Next" (Próximo)

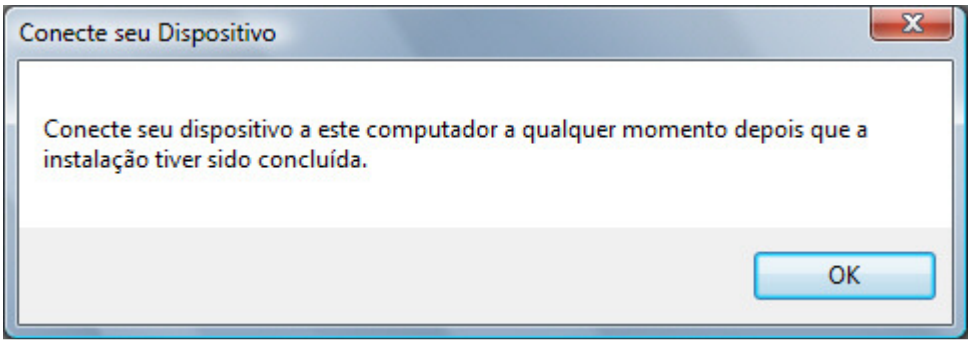


5.2.3) No formulário seguinte, confirme a instalação clicando no botão "Install" (Instalar).

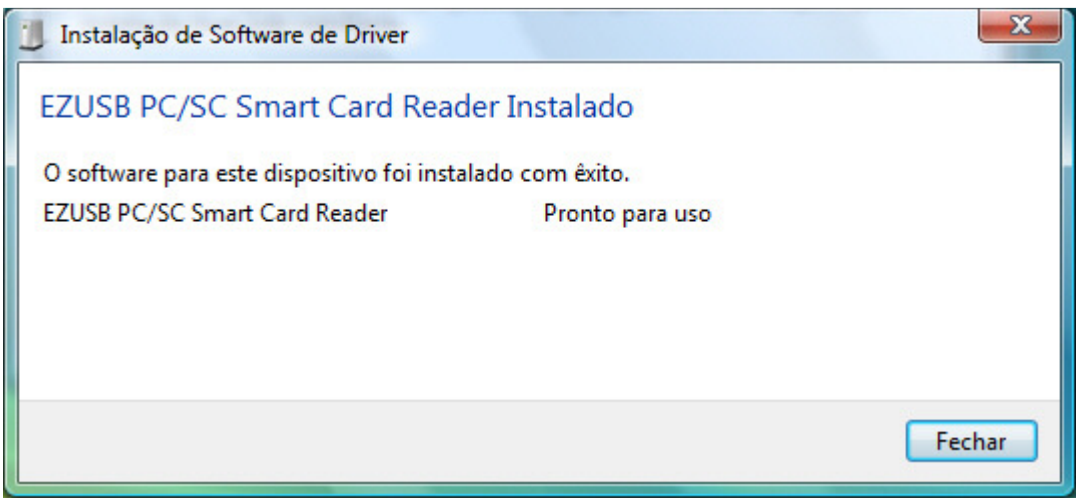




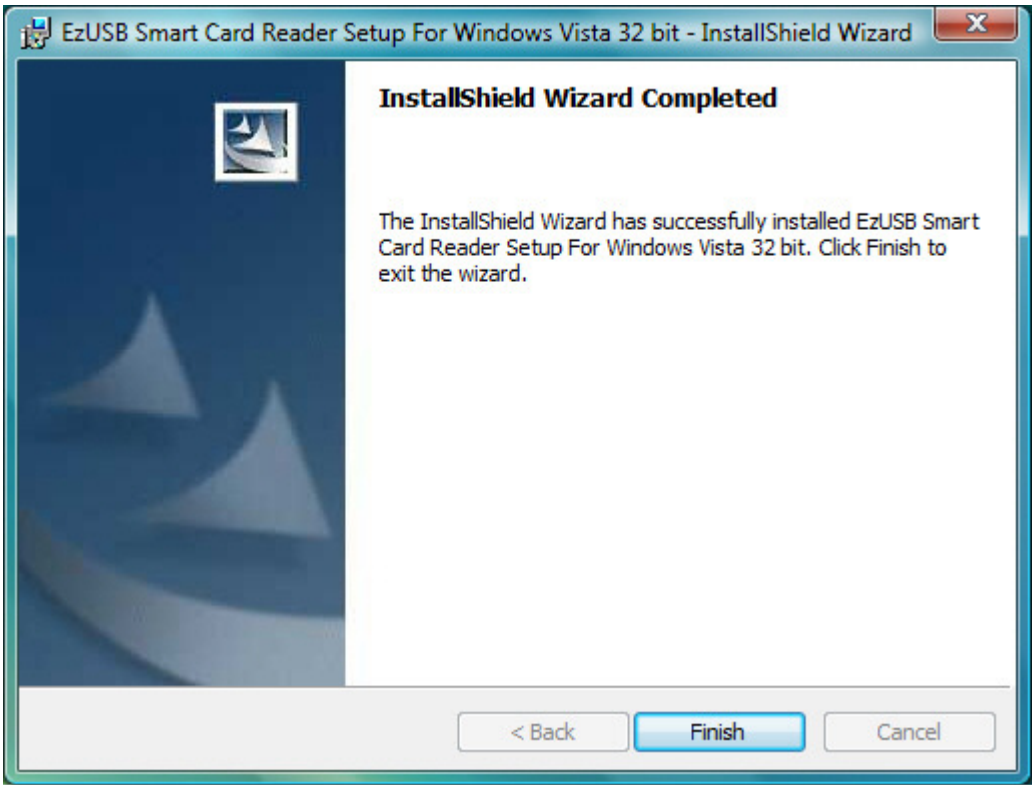
5.2.4) O programa de instalação copiará os arquivos do driver da leitora Smartnonus.
No final do processo de instalação, você receberá um aviso, neste instante **conecte** a leitora Smartnonus numa porta USB livre de seu computador, e clique no botão OK.



Quando conectar o leitor na porta USB, os drivers de instalação agora serão encontrados pelo Sistema Operacional.

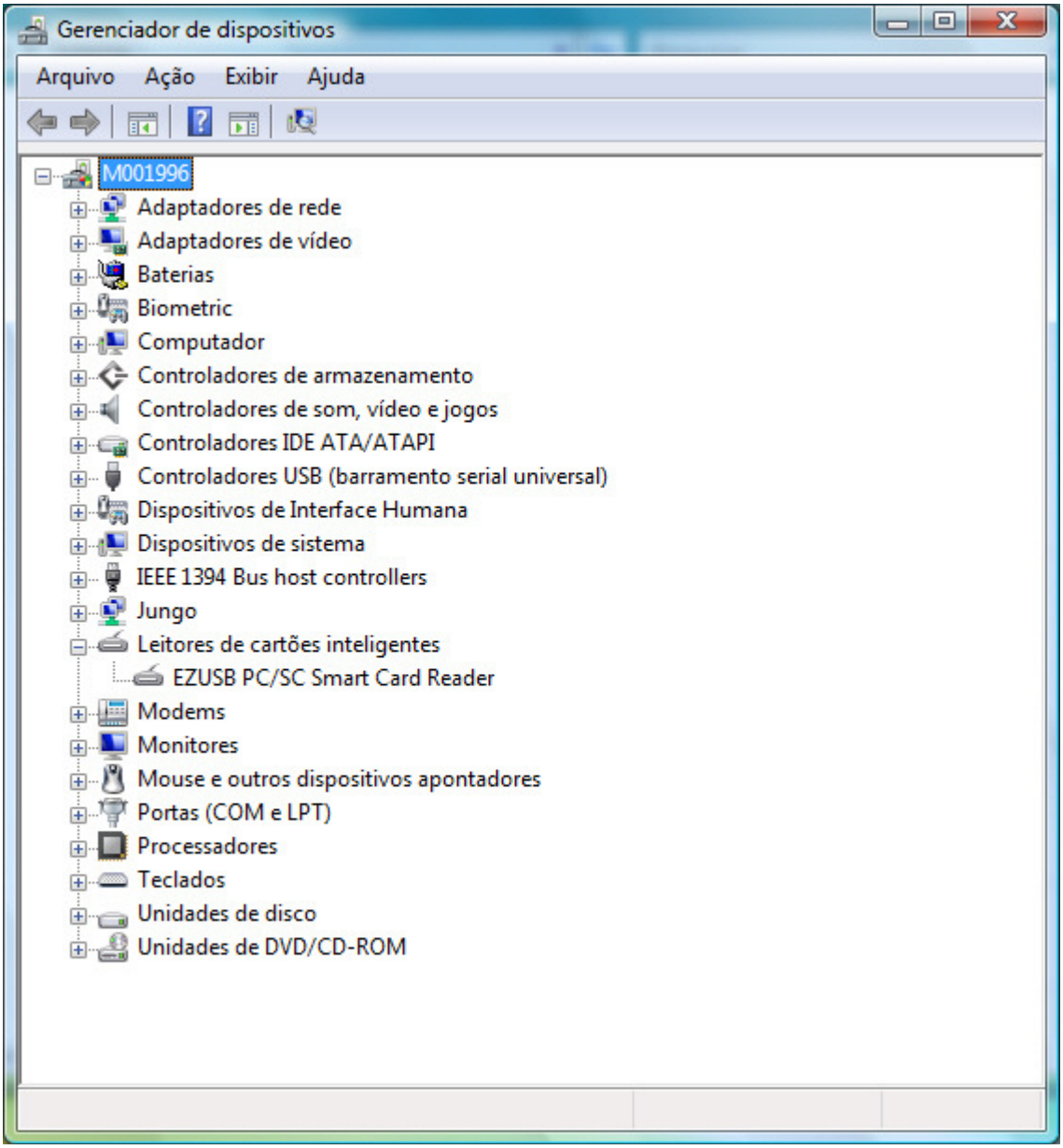


5.2.5) Por fim clique em "Finish" ou "Concluir".



5.2.6) Poderá ainda fazer uma verificação adicional de instalação bem sucedida.
Aponte e clique no Menu Iniciar, clique em Configurações, e em seguida clique em Painel de Controle.
Localize o ícone Sistema, aponte e clique em Gerenciador de Dispositivos.

Com o Gerenciador de Dispositivos aberto, clique para expandir o grupo Leitoras de Cartões Inteligentes.
Poderá visualizar a leitora Smartnonus descrita como EZUSB PC/SC Smart Card Reader.



5.3) Microsoft Windows XP

Requisitos Mínimos de Ambiente Recomendados:
Microsoft Windows XP (Service Pack 3)
Processador: 1GHz (ou superior)
Memória: Mínimo de 512MB
Espaço em Disco: 20MB
Versão do Driver: 3.1.6.0

Mantenha a leitora Smartnonus **desconectada** durante a instalação dos drivers para o Microsoft Windows XP.
Você precisará de direitos administrativos para prosseguir com a instalação dos drives para a leitora Smartnonus.

Os drivers mais atuais estão disponíveis para download na página da leitora Smartnonus na Internet em www.nonus.com.br/smartnonus.
Selecione o download do arquivo de driver para o Windows XP.

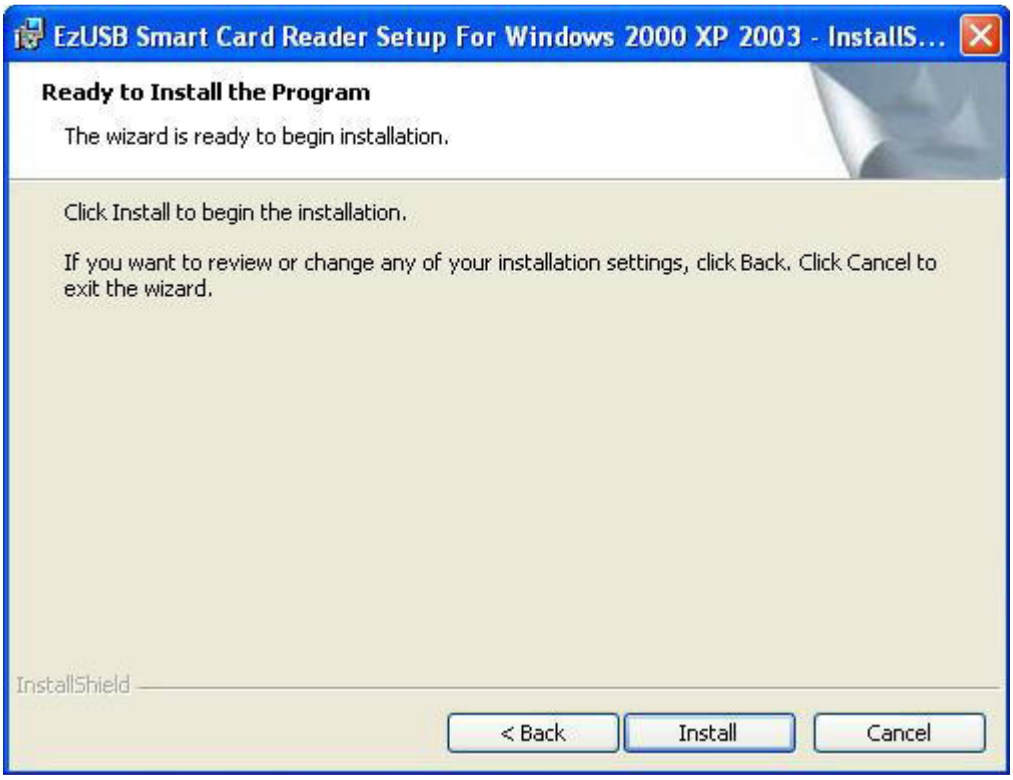
Quando o seu navegador concluir o download, poderá executar diretamente o programa de instalação ou salvar o arquivo numa pasta local, como por exemplo sua pasta "Meus Documentos".

5.3.1) Execute o arquivo de instalação a partir de uma pasta local ou da página da leitora Smartnonus.

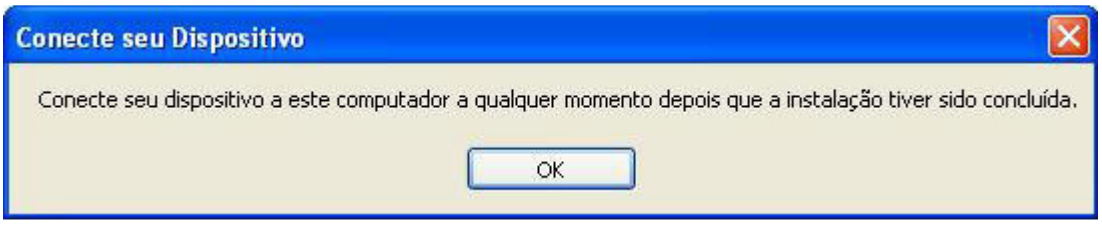
5.3.2) Na opção "Choose Setup Language" (Escolha a Linguagem de Configuração), escolha Português(Brasileiro), se estiver disponível, ou Inglês (Estados Unidos).

5.3.3) No formulário seguinte, aponte e clique no botão "Next" (Próximo)

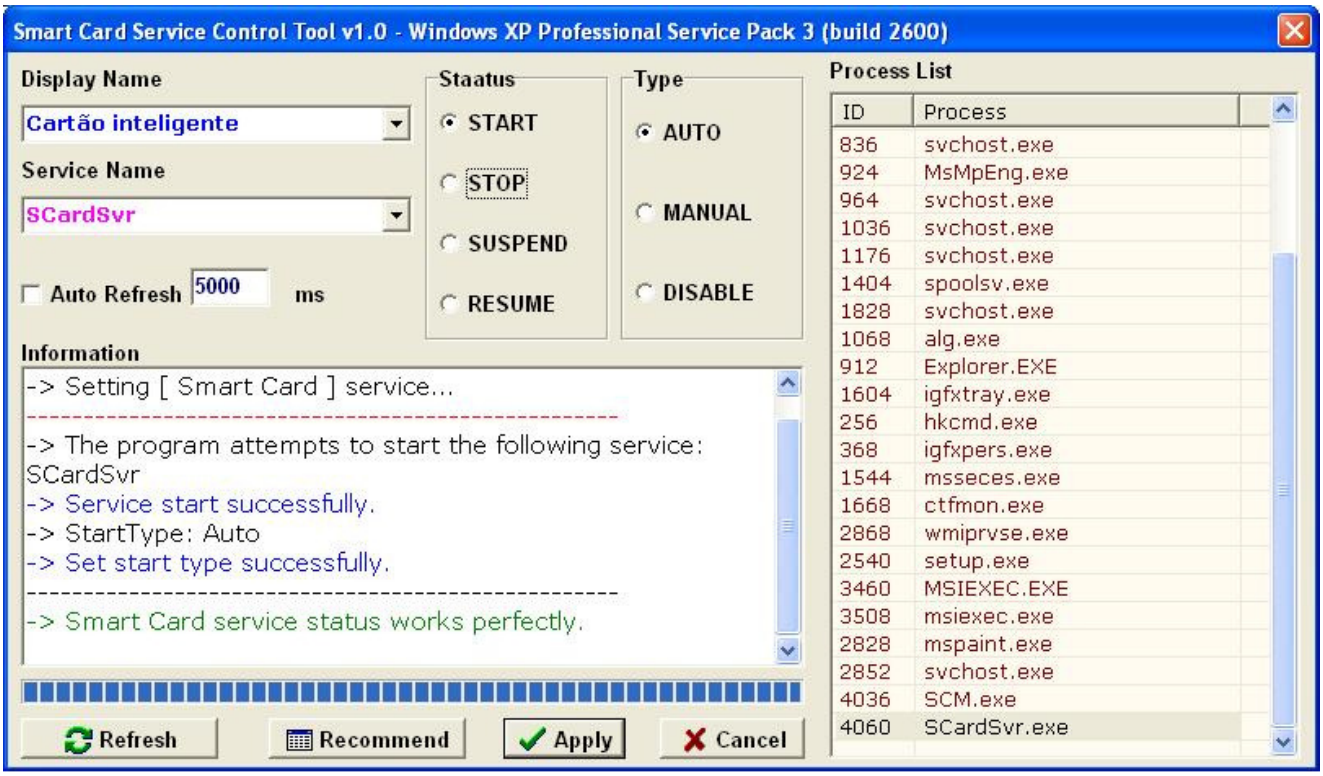
5.3.4) No formulário seguinte, confirme a instalação clicando no botão "Install" (Instalar).



5.3.5) O programa de instalação copiará os arquivos do driver da leitora Smartnonus.
No final do processo de instalação, você receberá um aviso, neste instante **conecte** a leitora Smartnonus numa porta USB livre de seu computador, e clique no botão OK.



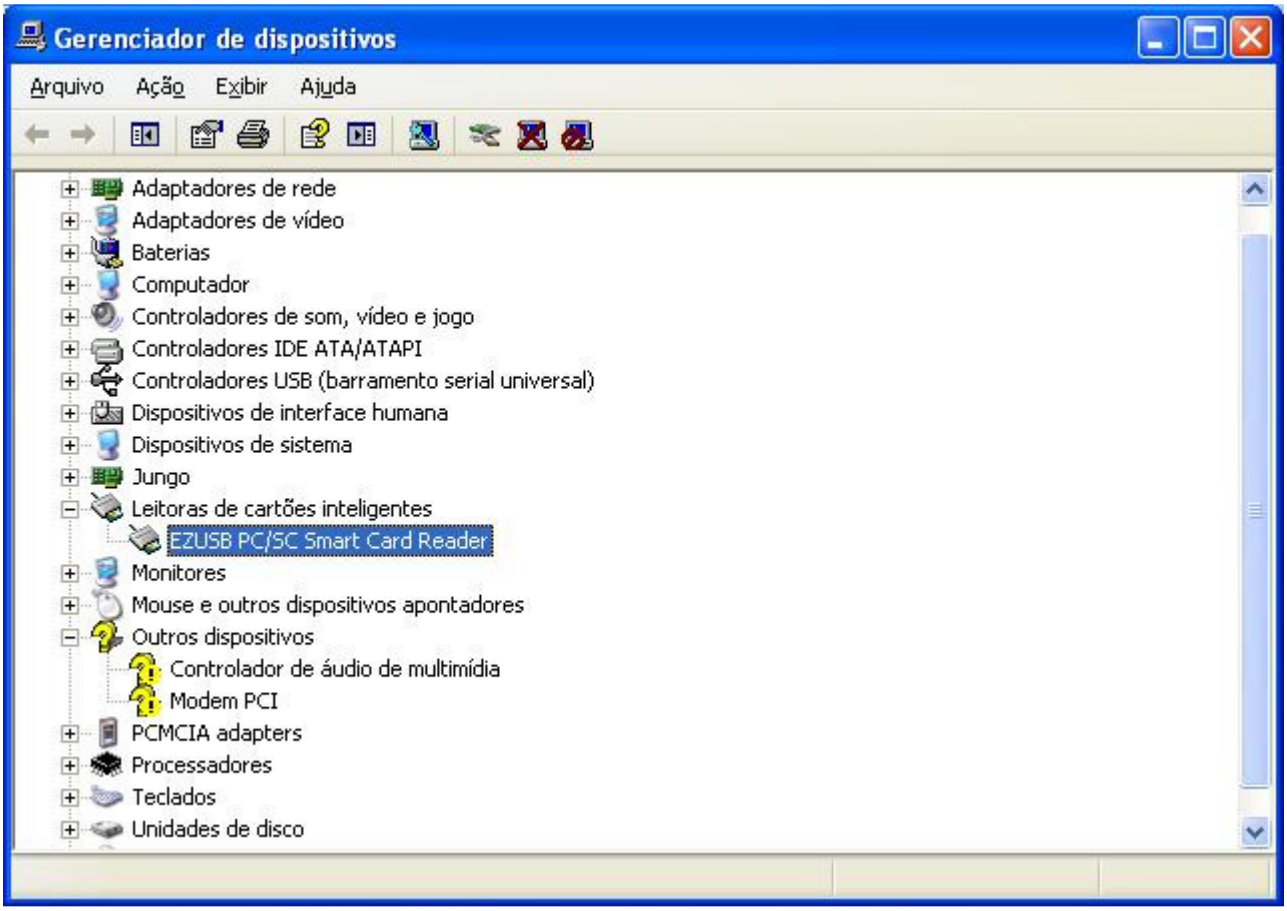
5.3.6) Caso o seu Windows XP esteja atualizado para a versão SP3 (Service Pack 3), o programa de instalação vai invocar o serviço de controle de cartão inteligente.
(Smart Card Service Tool v1.0). Clique no botão "Apply", em seguida feche a janela.



5.3.7) Para conferir a instalação bem sucedida, aponte e clique no menu "Iniciar", e em seguida clique em Painel de Controle, selecione e clique no ícone Sistema.

Na aba Hardware, clique no botão "Gerenciador de Dispositivos".

Com o Gerenciador de Dispositivos aberto, clique para expandir o grupo "Leitoras de Cartões Inteligentes" e poderá visualizar a leitora Smartnonus descrita como EZUSB PC/SC Smart Card Reader.



5.4) Linux

Requisitos Mínimos de Ambiente Recomendados:
Kernel 2.4 (ou superior)
Processador: 1GHz (ou superior)
Memória: Mínimo de 512MB
Espaço em Disco: 20MB
Versão do Driver: 1.4.9

O arquivo de driver mais atual para Linux está disponível na página da leitora Smartnonus na Internet em www.nonus.com.br/smartnonus. Selecione o download do arquivo de driver para Linux.
Você precisará de privilégios de root para instalar o driver da leitora Smartnonus.

Após o download, descompacte o arquivo Linux-x86.tar.
Após descompactado, este é o conteúdo do arquivo de driver, versão 1.4.9:

Pasta driver_ezusb_v1.4.9:
- check_env - Script para verificação do ambiente de instalação.
- install - Script de instalação
- driver_install - Sub Programa de instalação
- drivers/ezusb.so - Driver da leitora
- drivers/Info.plist - Arquivo de configuração do PCSCLITE USB

Pasta mifdtest:
- Contém um programa de teste PC/SC simplificado para identificar se o driver esta instalado e ativo.

Para efetuar a instalação complete os seguintes passos:

5.4.1) Execute o script de verificação do ambiente de instalação:

./check_env

Se o terminal exibir a mensagem: "PC/SC Daemon Not Ready", por favor vá para o site:
http://www.linuxnet.com/middle.html para fazer o download e instalar o pacote pcsclite.

Se o terminal exibir a mensagem "Error ! USB Device File System Not Mounted", por favor monte o sistema de arquivo USB de acordo com a instrução.

Observação: Caso o script não possa ser executado, por favor utilize "chmod 777 check_env" para alterar o script para um arquivo executável.

5.4.2) Execute o script de instalação: ./install

Observação: Caso o script não possa ser executado, por favor utilize "chmod 777 install" para alterar o script para um arquivo executável. Lembre-se que você precisará de privilégios de root.

5.4.3) Reinicie o sistema

5.4.4) Após reiniciar, insira um cartão. Se o led da leitora ascender na cor vermelha, a instalação da leitora foi bem sucedida.

Outras observações:
a) A versão recomendada do Kernel é 2.4 ou maior.
b) Se você tiver instalado uma versão anterior do driver da Smartnonus (v. 1.3.4 ou menor) por favor desinstale primeiro antes de executar uma nova instalação.
c) O driver requer que PCSCLITE tenha sido compilado com libusb. Se PCSCLITE foi compilado com libhal, o driver não irá funcionar.

5.5) Mac OSX

Requisitos Mínimos de Ambiente Recomendados:
MacMini CoreDuo
Mac OSx 10.5.x (ou superior)
Memória: Mínimo de 1GB
Espaço em Disco: 200MB
Versão do Driver: 1.4.1

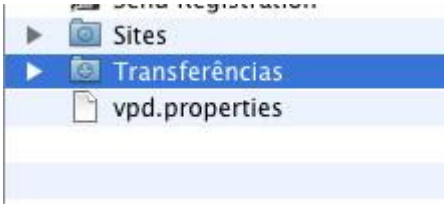
Os drivers mais atuais estão disponíveis na página da leitora Smartnonus na Internet em www.nonus.com.br/smartnonus. Selecione o arquivo para o MAC OSX (macosx.zip).

Recomendamos também a instalação do Token Admin, disponível em:
http://www.certisign.com.br/downloads/SafeSign3.0_MAC10.5_Leopard.zip

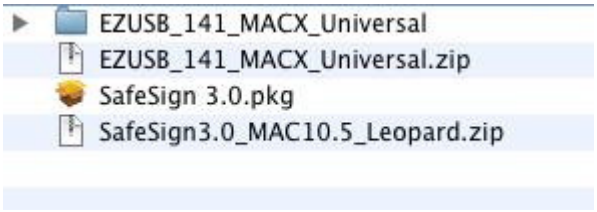
5.5.1) Depois da transferência dos arquivos acima, clique no Finder:

5.5.2) Clique no menu da parte superior da tela e escolha "Início":

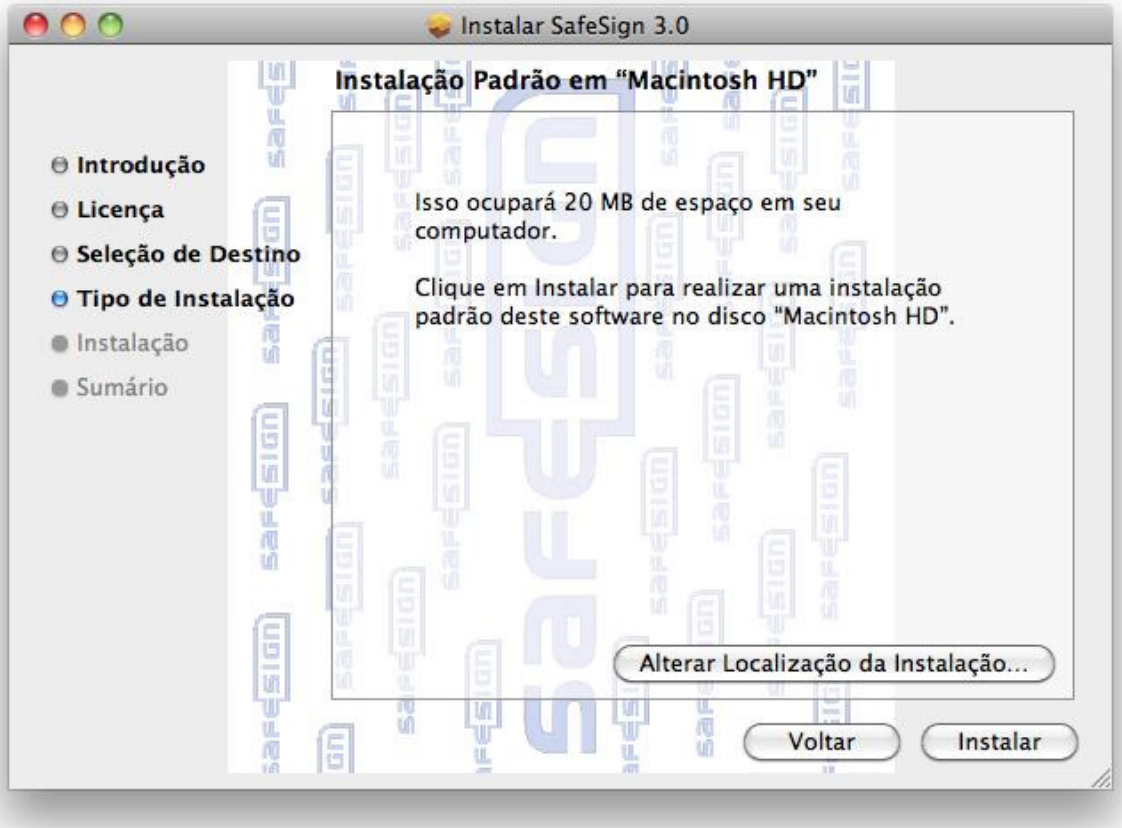
5.5.3) Clique na pasta de "Transferências".
Obs: Em alguns MACs existe um atalho para transferências ao lado do ícone do lixo.

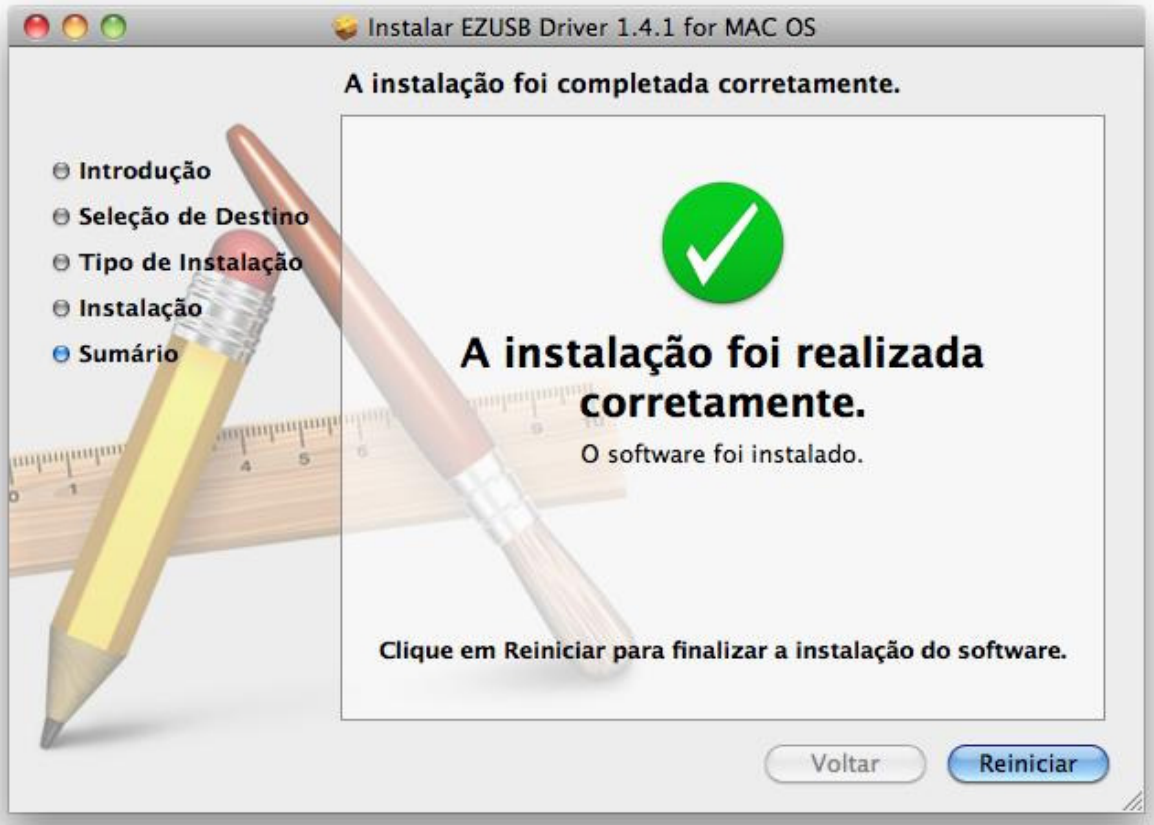


5.5.4) Na pasta "Transferências", você encontrará os arquivos, conforme figura abaixo.
Inicie com a instalação do SafeSign, execute o arquivo SafeSign, e confirme a instalação até o final.



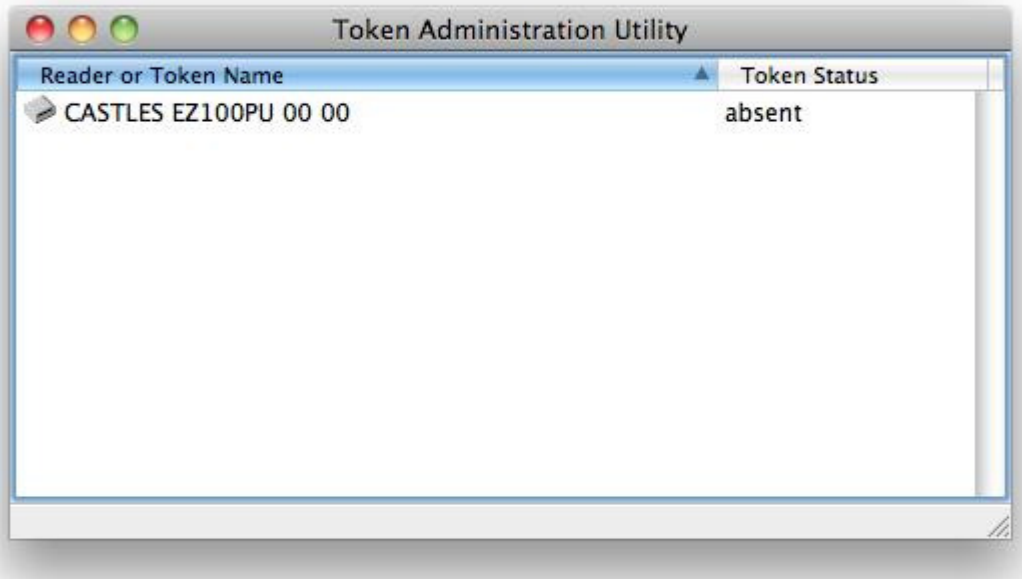
5.5. 5) Em seguida execute o mesmo procedimento, abra a pasta "EZUSB_141_MACX_Universal" e execute o arquivo de driver "ezusb_driver_setup.mpkg".





5.5.6) Reinicie o MAC.
Em seguida conecte a leitora Smartnonus.
Clique no Menu "IR", em seguida em "Aplicativos", encontre e abra o programa "TokenAdmin".

Neste programa, você poderá verificar a leitora Smartnonus instalada, e se inserir um cartão, na coluna "Token Status", verificará a mensagem "Present".



6) Referência API PC/SC - Smartnonus

6.1) Visão Geral da API

A documentação abaixo descreve as funções presentes na API da leitora Smartnonus.
Nesta primeira parte estão agrupadas por funcionalidade, em seguida apresentamos cada função individualmente.

A Nonus fornece também uma demonstração da API em funcionamento com o código fonte.
As funções principais estão exemplificadas nesta demonstração, bem como os procedimentos de conexão e recuperação de dados da leitora.
A demonstração da API Smartnonus pode ser obtida no site da Nonus em www.nonus.com.br/smartnonus, para mais informações consulte o Apêndice C - Demonstração API Smartnonus v.1.0.

6.1.1) Funções de Consulta à Base de Dados do Cartão Inteligente

Estas funções consultam a base de dados do cartão inteligente. Elas podem fornecer uma lista de cartões inteligentes fornecidos por um usuário específico, as interfaces e os fornecedores de serviços primários de um cartão específico, os grupos de leitoras definidas para o sistema, e as leitoras a partir de um conjunto de grupos de leitoras.
Ao utilizar estas funções, você pode consultar a base de dados completa do cartão inteligente ou reduzir a busca definindo o contexto do gerente de recursos. O contexto do gerente de recursos é definido selecionando CasEstablishContext() antes de selecionar uma função de consulta.
Nota: Sem contexto específico, algumas informações podem ainda ser inacessíveis devido às restrições de segurança.

Para…	Selecione…
Recuperar o identificador (GUID) do fornecedor do serviço primário para o cartão dado.	CasGetProviderId
Recuperar uma lista de cartões introduzida previamente no sistema por um usuário específico.	CasListCards
Recuperar os identificadores (GUID) das interfaces fornecidas para um cartão dado.	CasListInterfaces
Recuperar uma lista de grupos de leitoras que foi introduzida previamente no sistema.	CasListReaderGroups
Recuperar a lista de leitoras dentro de um conjunto de grupos de leitoras designadas.	CasListReaders

6.1.2) Funções de Gerenciamento da Base de Dados do Cartão Inteligente

Estas funções gerenciam a base de dados do cartão inteligente, atualizando a base de dados utilizando um contexto do gerente de recursos especificado.
Nota: A segurança da base de dados é mantida estabelecendo restrições de acesso à base de dados ao invés de adicionar mecanismos de segurança ao subsistema do cartão inteligente.

Para…	Selecione…
Adicionar uma leitora ao grupo de leitoras.	CasAddReaderToGroup
Remover um cartão inteligente do sistema.	CasForgetCardType
Remover uma leitora do sistema.	CasForgetReader
Remover um grupo de leitoras do sistema.	CasForgetReaderGroup
Introduzir um novo cartão ao sistema.	CasIntroduceCardType
Introduzir uma nova leitora ao sistema.	CasIntroduceReader
Introduzir um novo grupo de leitoras ao sistema.	CasIntroduceReaderGroup
Remover uma leitora de um grupo de leitoras.	CasRemoveReaderFromGroup

6.1.3) Funções de Acesso Direto do Cartão

O subsistema do cartão inteligente permite que você se comunique com cartões que podem não estar em conformidade com as especificações da ISO 7816. Para fazer isto, estas funções permitem que você controle os atributos das comunicações entre o aplicativo e o cartão, dando a você a possibilidade de manipulação direta, de baixo nível, da leitora.

Para…	Selecione…
Fornecer controle direto da leitora.	CasControl
Obter atributos da leitora.	CasGetAttrib
Estabelecer atributos da leitora.	CasSetAttrib

6.1.4) Funções do Contexto do Gerente de Recursos

Estas funções estabelecem e liberam o contexto do gerente de recursos que é utilizado pela consulta à base de dados e às funções de gerenciamento da base de dados.

Para…	Selecione…
Estabelecer um contexto para acessar a base de dados do cartão inteligente.	CasEstablishContext
Fechar um contexto estabelecido.	CasReleaseContext

6.1.5) Função de Suporte do Gerente de Recursos

Esta função libera a memória alocada através do uso do designador de comprimento SCARD_AUTOALLOCATE, simplificando o uso das outras funções do gerente de recursos.

Para…	Selecione…
Liberar memória retornada através do uso do SCARD_AUTOALLOCATE.	CasFreeMemory

6.1.6) Funções de Rastreabilidade do Cartão Inteligente

Estas funções permitem que você rastreie cartões em leitoras. Estas rotinas normalmente utilizam a estrutura SCARD_READERSTATE dentro de uma matriz.

Para…	Selecione…
Procurar um cartão cuja sequência de ATR (atributos) corresponda a um nome de cartão fornecido.	CasLocateCards
Bloquear a execução até que a disponibilidade atual de cartões mude.	CasGetStatusChange
Concluir ações pendentes.	CasCancel

6.1.7) Funções de Acesso do Cartão Inteligente e da Leitora

Estas funções se conectam e comunicam com um cartão inteligente específico. Operações de entrada/saída do cartão utilizam um buffer para envio e recebimento de dados, e uma estrutura que contém informações para controle de protocolos. A estrutura de informações para controle de protocolo se inicia sempre com uma estrutura SCARD_IO_REQUEST.

Para…	Selecione…
Conectar um cartão.	CasConnect
Restabelecer uma conexão.	CasReconnect
Concluir uma conexão.	CasDisconnect
Iniciar uma transação, bloqueando o acesso de outros aplicativos ao cartão.	CasBeginTransaction
Terminar uma transação, permitindo o acesso de outros aplicativos ao cartão.	CasEndTransaction
Fornecer o status atual da leitora.	CasStatus
Solicitar serviço e receber dados de volta de um cartão utilizando T=0, T=1, e protocolos RAW.	CasTransmit

6.2) Estrutura do Cartão Inteligente

6.2.1) SCARD_IO_REQUEST

A estrutura SCARD_IO_REQUEST inicia uma estrutura de informações para controle de protocolo. Então, qualquer informação específica de protocolo imediatamente segue esta estrutura. O comprimento inteiro da estrutura deve estar alinhado com o tamanho de palavras da arquitetura de hardware subjacente. Por exemplo, em Win32 o comprimento de qualquer informação de PCI deve ser múltiplo de 4 bytes para que ele se alinhe com uma vizinhança de 32 bits.

```
typedef struct {
        DWORD    dwProtocol
        DWORD    cbPciLength;
} SCARD_IO_REQUEST;
```

Membros

dwProtocol
Identifica o protocolo em uso.

cbPciLength
Fornece o comprimento, em bytes, da estrutura SCARD_IO_REQUEST, mais qualquer informação específica de PCI seguinte

6.2.2) SCARD_READERSTATE

A estrutura SCARD_READERSTATE é utilizada pelas funções para rastrear cartões inteligentes em leitoras.

```
typedef struct {
                    LPCTSTR   szReader
                    LPVOID    pvUserData;
                    DWORD     dwCurrentState;
                    DWORD     dwEventState;
                    DWORD     cbAtr;
                    BYTE      rgbAtr[36];
} SCARD_READERSTATE, *PSCARD_READERSTATE, *LPSCARD_READERSTATE;
```

Membros

szReader
Aponta o nome da leitora que está sendo monitorada.

pvUserData
Não utilizado pelo subsistema do cartão inteligente. Utilizado pelo aplicativo.

dwCurrentState
Fornece o status atual da leitora, conforme visto pelo aplicativo. Este campo pode assumir qualquer um dos valores a seguir, em combinação, como uma máscara de bits:

Valor	Significado
SCARD_STATE_UNAWARE	O aplicativo desconhece o status presente, e gostaria de saber. O uso deste valor resulta em um retorno imediato dos serviços de monitoramento de transição de status. Isto é representado por todos os bits sendo zerados.
SCARD_STATE_IGNORE	O aplicativo não está interessado nesta leitora, e isto deve ser considerado durante as operações de monitoramento. Se este valor de bit for ajustado, todos os outros bits serão ignorados.
SCARD_STATE_UNAVAILABLE	O aplicativo acredita que esta leitora não está disponível para uso. Se este bit for ajustado, então todos os bits seguintes serão ignorados.
SCARD_STATE_EMPTY	O aplicativo acredita que não há cartão na leitora. Se este bit for ajustado, todos os bits seguintes serão ignorados.
SCARD_STATE_PRESENT	O aplicativo acredita que há um cartão na leitora.
SCARD_STATE_ATRMATCH	O aplicativo acredita que há um cartão na leitora com um ATR correspondendo a um dos cartões alvo. Se este bit for ajustado, SCARD_STATE_PRESENT é assumido. Este bit não tem nenhum significado para o CasGetStatusChange além de SCARD_STATE_PRESENT.
SCARD_STATE_EXCLUSIVE	O aplicativo acredita que o cartão na leitora está alocado para uso exclusivo por outro aplicativo. Se este bit for ajustado, SCARD_STATE_PRESENT é assumido.
SCARD_STATE_INUSE	O aplicativo acredita que o cartão na leitora está em uso exclusivo por um ou mais aplicativos, mas pode estar conectada em modo compartilhado. Se este bit for ajustado, SCARD_STATE_PRESENT é assumido.
SCARD_STATE_MUTE	O aplicativo acredita que há um cartão que não responde na leitora.

dwEventState
Recebe o status atual da leitora, conforme conhecido pelo gerente de recursos do cartão inteligente. Este campo pode assumir qualquer um dos valores a seguir, em combinação, como uma máscara de bits:

Valor	Significado
SCARD_STATE_IGNORE	Esta leitora deve ser ignorada.
SCARD_STATE_CHANGED	Há uma diferença entre o status reconhecido pelo aplicativo e o status conhecido pelo gerente de recursos. Quando este bit é ajustado, o aplicativo pode assumir uma mudança significativa de status ocorrida nesta leitora.
SCARD_STATE_UNKNOWN	Esta leitora dada não é reconhecida por este gerente de recursos. Se este bit for ajustado, então SCARD_STATE_CHANGED e SCARD_STATE_IGNORE também serão ajustados.
SCARD_STATE_UNAVAILABLE	O status real desta leitora não está disponível. Se este bit for ajustado, então todos os bits seguintes estarão sem código.
SCARD_STATE_EMPTY	Não há cartão na leitora. Se este bit for ajustado, todos os bits seguintes estarão sem código.
SCARD_STATE_PRESENT	Há um cartão na leitora.
SCARD_STATE_ATRMATCH	Há um cartão na leitora com um ATR correspondendo a um dos cartões alvo. Se este bit for ajustado, SCARD_STATE_PRESENT também será ajustado. Este bit apenas retorna na função CasLocateCards.
SCARD_STATE_EXCLUSIVE	O cartão na leitora está alocado para uso exclusivo por outro aplicativo. Se este bit for ajustado, SCARD_STATE_PRESENT também será ajustado.
SCARD_STATE_INUSE	O cartão na leitora está em uso exclusivo por um ou mais aplicativos, mas pode estar conectada em modo compartilhado. Se este bit for ajustado, SCARD_STATE_PRESENT também será ajustado.
SCARD_STATE_MUTE	Há um cartão que não responde na leitora.

cbAtr
O número de bytes no ATR retornado.

rgbAtr
O ATR do cartão inserido, com bytes de alinhamento extras.

6.2.3) OPENCARDNAME_EX

A estrutura OPENCARDNAME_EX contém a informação que a função **CasUIDlgSelectCard** utiliza para inicializar uma caixa de diálogo Select Card de um cartão inteligente.

```
typedef struct {
        DWORD           dwStructSize;
        SCARDCONTEXT    hSCardContext;
        HWND            hwndOwner;
        DWORD           dwFlags;
        LPCTSTR         lpstrTitle;
        LPCTSTR         lpstrSearchDesc;
        HICON           hIcon;
        POPENCARD_SEARCH_CRITERIA       pOpenCardSearchCriteria;
        LPOCNCONNPROC           lpfnConnect;
        LPVOID          pvUserData;
        DWORD           dwShareMode;
        DWORD           dwPreferredProtocols;
        LPTSTR          lpstrRdr;
        DWORD           nMaxCard;
        DWORD           dwActiveProtocol;
        SCARDHANDLE     hCardHandle;
} OPENCARDNAME_EX, *POPENCARDNAME_EX, *LPOPENCARDNAME_EX;
```

Membros

dwStructSize
Especifica o comprimento, em bytes, da estrutura. O valor deste membro não deverá ser NULL.

hSCardContext
Contexto utilizado para comunicação com o gerente de recursos do cartão inteligente. Selecione CasEstablishContext para ajustar o contexto do gerente de recursos e CasReleaseContext para liberá-lo. O valor deste membro não deverá ser NULL.

hwndOwner
Identifica a janela que possui a caixa de diálogo. Este membro poderá ser qualquer identificador de janela válido, ou pode ser NULL para o padrão de desktop.

dwFlags
Um conjunto de bit flags (marcadores em forma de bit) que você pode utilizar para inicializar a caixa de diálogo. Quando a caixa de diálogo retornar, ela ajusta estes marcadores para indicar a entrada do usuário. Este membro poderá ser um dos marcadores a seguir:

SC_DLG_MINIMAL_UI Mostra o diálogo apenas se o cartão a ser procurado pelo aplicativo de seleção não foi localizado e estiver disponível para uso em uma leitora. Isto permite que o cartão seja encontrado, conectado (através do mecanismo interno de diálogo ou das funções de chamada do usuário), e retornado ao aplicativo de seleção.
SC_DLG_NO_UI Força para que a Interface de Usuário (IU) do Select Card não seja exibida, independente do resultado da pesquisa.
SC_DLG_NO_UI Força para que a Interface de Usuário (IU) do Select Card seja exibida, independente do resultado da pesquisa.

lpstrTitle
Aponta para uma sequência a ser colocada na barra de título da caixa de diálogo. Se este membro for NULL, o sistema utiliza o título padrão “Selecionar Cartão:”.

lpstrSearchDesc
Aponta para uma sequência a ser exibida para o usuário como uma solicitação para inserir o cartão inteligente. Se este membro for NULL, o sistema utiliza o título padrão “Favor inserir um cartão inteligente”.

hIcon
Identificador para um ícone (32 x 32 pixels). Você pode especificar um ícone específico para um fornecedor, para ser exibido no diálogo. Se este valor for NULL, um ícone genérico gerado pela leitora do cartão inteligente é exibido.

pOpenCardSearchCriteria
Apontador para a estrutura OPENCARD_SEARCH_CRITERIA a ser utilizada, ou NULL se uma não for utilizada.

lpfnConnect
Apontador para a rotina de conexão do cartão do selecionador. Se o selecionador precisa realizar um processamento adicional para se conectar ao cartão, este apontador de função é ajustado para a função de conexão do usuário. Se a função de conexão for bem sucedida, o cartão é deixado conectado e inicializado, e o identificador do cartão é retornado. O protótipo para a rotina de conexão é dada a seguir:

```
Connect(
        hSCardContext,  // o contexto do cartão passado no bloco do parâmetro
        szReader,       // o nome da leitora
        mszCards,       // uma sequência que contém os nomes de cartão possíveis na leitora
        pvUserData      // apontador para dados do usuário passados no bloco do parâmetro
       );
```

pvUserData
Um apontador inválido para dados do usuário. Este apontador é passado de volta para o selecionador na rotina de conexão.

dwShareMode
Se lpfnConnect não for NULL, os membros dwShareMode e dwPreferredProtocols serão ignorados. Se lpfnConnect for NULL e dwShareMode for diferente de zero, uma seleção interna será feita para CasConnect usando dwShareMode e dwPreferredProtocols como os parâmetros dwShareMode e dwPreferredProtocols. Se a conexão for bem sucedida, hCardHandle é ajustado para o identificador retornado pelo CasConnect. Se lpfnConnect for NULL e dwShareMode for zero, hCardHandle é ajustado para NULL.

dwPreferredProtocols
Usado para conexão interna, conforme descrito em dwShareMode.

lpstrRdr
Se o cartão for localizado, o buffer lpstrRdr contém o nome da leitora que contém o cartão localizado. O buffer deve ter o comprimento de 256 caracteres, no mínimo.

nMaxRdr
Especifica o tamanho, em bytes (versão ANSI) ou caracteres (versão UNICODE), do buffer apontado por lpstrRdr. Se o buffer for pequeno demais para conter a informação da leitora, CasUIDlgSelectCard retorna SCARD_E_NO_MEMORY e o tamanho necessário do buffer é apontado por lpstrRdr.

lpstrCard
Se o cartão for localizado, o buffer lpstrRdr contém o nome do cartão localizado. O buffer deve ter o comprimento de 256 caracteres, no mínimo.

nMaxCard
Especifica o tamanho, em bytes (versão ANSI) ou caracteres (versão UNICODE), do buffer apontado por lpstrRdr. Se o buffer for pequeno demais para conter a informação do cartão, CasUIDlgSelectCard retorna SCARD_E_NO_MEMORY e o tamanho necessário do buffer em nMaxCard.

dwActiveProtocol
Retorna o protocolo em uso quando o diálogo faz uma conexão com um cartão.

hCardHandle
Identificação do cartão conectado (seja através de uma conexão de diálogo interno ou de uma chamada lpfnConnect).

6.2.4) OPENCARD_SEARCH_CRITERIA

A estrutura OPENCARD_SEARCH_CRITERIA é utilizada pela função CasUIDlgSelectCard a fim de reconhecer cartões que tenham os requisitos necessários estabelecidos pelo selecionador.
No entanto, você pode selecionar CasUIDlgSelectCard sem utilizar esta estrutura.

```
typedef struct {
                    DWORD                   dwStructSize;
                    LPTSTR                  lpstrGroupNames;
                    DWORD                   nMaxGroupNames;
                    LPCGUID                 rgguidInterfaces;
                    DWORD                   cguidInterfaces;
                    LPTSTR                  lpstrCardNames;
                    DWORD                   nMaxCardNames;
                    LPOCNCHKPROC            lpfnCheck;
                    LPOCNCONNPROC           lpfnConnect;
                    LPOCNDSCPROC            lpfnDisconnect;
                    LPVOID                  pvUserData;
                    DWORD                   dwShareMode;
                    DWORD                   dwPreferredProtocols;
} OPENCARD_SEARCH_CRITERIA, *POPENCARD_SEARCH_CRITERIA,
*LPOPENCARD_SEARCH_CRITERIA;
```

Membros:
dwStructSize
Especifica o comprimento, em bytes, da estrutura. Não deverá ser NULL.

lpstrGroupNames
Aponta para um buffer que contenha sequências de nomes de grupos terminados em NULL. A última sequência no buffer deve ser terminada por dois caracteres NULL. Cada sequência é o nome de um grupo de cartão que deve ser incluído na busca. Se lpstrGroupNames for NULL, o grupo padrão (Scard$DefaultReaders) é buscado.

nMaxGroupNames
Número máximo de bytes (versão ANSI) ou caracteres (versão UNICODE) na sequência lpstrGroupNames.

rgguidInterfaces
Reservado para uso futuro. Uma matriz de GUIDs que identificam as interfaces necessárias. Ajuste este membro para NULL para esta liberação.

cguidInterfaces
Reservado para uso futuro. O número de interfaces na matriz rgguidInterfaces. Ajuste este membro para NULL para esta liberação.

lpstrCardNames
Aponta para um buffer que contenha sequências de nomes de grupos terminados em NULL. A última sequência no buffer deve ser terminada por dois caracteres NULL. Cada sequência é o nome de um cartão que deve ser localizado.

nMaxCardNames
Número máximo de bytes (versão ANSI) ou caracteres (versão UNICODE) na sequência lpstrCardNames.

lpfnCheck
Aponta para a rotina de verificação do cartão do selecionador. Se nenhuma verificação especial de cartão for necessária, este apontador é NULL. Se o cartão for rejeitado pela rotina de verificação, é retornado FALSE e o cartão será desconectado. Se o cartão for rejeitado pela rotina de verificação, é retornado TRUE.

O protótipo para a rotina de verificação é:

```
Boolean Check(
        SCardContext,    // o contexto do cartão passado no bloco do parâmetro
        hCard,           // identificador do cartão
        pvUserData       // apontador para dados do usuário passados no bloco do parâmetro
        );
```

lpfnConnect
Aponta para a rotina de conexão do cartão do selecionador. Se o selecionador precisa realizar um processamento adicional para se conectar ao cartão, este apontador de função é ajustado para a função de conexão do usuário. Se a função de conexão for bem sucedida, o cartão é deixado conectado e inicializado, e o identificador do cartão é retornado. O protótipo para a rotina de conexão é:

```
Connect(
        hSCardContext,   // o contexto do cartão passado no bloco do parâmetro
        szReader,        // o nome da leitora
        mszCards,        // uma multi sequência que contém os nomes de cartão possíveis na leitora
        pvUserData       // apontador para dados do usuário passados no bloco do parâmetro
        );
```

lpfnDisconnect
Aponta para a rotina de desconexão do cartão do selecionador.
O protótipo para a rotina de desconexão é:

```
Disconnect(
        hSCardContext,   // o contexto do cartão passado no bloco do parâmetro
        hCard,   // identificador do cartão
        pvUserData       // apontador para dados do usuário passados no bloco do parâmetro
        );
```

Nota: Quando você utilizar lpfnConnect, lpfnCheck, e lpfnDisconnect, todos os três procedimentos de chamada devem estar presentes. A utilização destas chamadas permite posterior verificação sobre se o aplicativo de seleção encontrou o cartão apropriado. Esta é a melhor maneira de garantir que o cartão apropriado foi selecionado. No entanto, ao utilizar um valor não NULL para lpfnCheck, ou lpfnConnect e lpfnDisconnect devem ser não NULL (e pvUserData também deve ser fornecido), ou dwShareMode e dwPreferredProtocols devem ser ajustados.

pvUserData
Um apontador inválido para dados do usuário. Este apontador é passado de volta para o selecionador nas rotinas de Conexão Verificação e Desconexão.

dwShareMode
Se lpfnConnect não for NULL, os membros dwShareMode e dwPreferredProtocols serão ignorados. Se lpfnConnect for NULL e dwShareMode for diferente de zero, uma seleção interna será feita para CasConnect usando dwShareMode e dwPreferredProtocols como parâmetros.

dwPreferredProtocols
Usado para conexão interna, conforme descrito em dwShareMode.

6.3) Referência da API

6.3.1) CasAddReaderToGroup()

A função CasAddReaderToGroup adiciona uma leitora ao grupo de leitoras.

```
LONG CasAddReaderToGroup(
        IN SCARDCONTEXT hContext,
        IN LPCTSTR szReaderName,
        IN LPCTSTR szGroupName
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia pelo comando CasEstablishContext.

szReaderName
Fornece o nome adequado da leitora que você está adicionando.

szGroupName
Fornece o nome adequado do grupo para o qual você está adicionando a leitora.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	–SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasAddReaderToGroup cria automaticamente o grupo de leitoras especificado, se ele ainda não existir.
CasAddReaderToGroup é uma função de gerenciamento da base de dados.

6.3.2) CasBeginTransaction()

A função CasBeginTransaction inicia uma transação, esperando pela conclusão de todas as outras transações antes que ela comece.

Quando a transação se iniciar, o acesso de todos os outros aplicativos ao cartão inteligente será bloqueado enquanto a transação estiver em curso.

```
LONG CasBeginTransaction(
                IN SCARDHANDLE hCard
);
```

Parâmetros
hCard
Fornece o valor de referência obtido de uma seleção anterior para o CasConnect.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasBeginTransaction é uma função de acesso do cartão inteligente e da leitora.

6.3.3) CasCancel()

A função CasCancel conclui todas as ações pendentes dentro de um contexto específico do gerente de recursos.

As únicas solicitações que você pode cancelar são aquelas que exigem que se espere por uma ação externa do cartão inteligente ou do usuário. Quaisquer solicitações por ações pendentes terminarão com uma indicação de status de que a ação foi cancelada. Isto é especialmente útil para forçar a conclusão de seleções pendentes de CasGetStatusChange.

```
LONG CasCancel(
        IN SCARDCONTEXT hContext
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasCancel é uma função de rastreabilidade do cartão inteligente.

6.3.4) CasConnect()

A função CasConnect estabelece uma conexão (utilizando um contexto específico do gerente de recursos) entre o aplicativo de seleção e o cartão inteligente contido por uma leitora específica. Se não houver cartão na leitora especificada, um erro é retornado.

```
LONG CasConnect(
        IN SCARDCONTEXT hContext,
        IN LPCTSTR szReader,
        IN DWORD dwShareMode,
        IN DWORD dwPreferredProtocols,
        OUT LPSCARDHANDLE phCard,
        OUT LPDWORD pwdActiveProtocol
);
```

Parâmetros

hContext

Fornece o identificador do contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext.

szReader

Fornece o nome da leitora que contém o cartão alvo.

dwShareMode

Fornece um marcador que indica se outros aplicativos podem formar conexões com o cartão. Possíveis valores:

Valor	Significado
SCARD_SHARE_SHARED	Este aplicativo deseja compartilhar o cartão com outros aplicativos.
SCARD_SHARE_EXCLUSIVE	Este aplicativo não deseja compartilhar o cartão com outros aplicativos.
SCARD_SHARE_DIRECT	Este aplicativo está alocando a leitora para seu uso privado, e o controlará diretamente. Nenhum outro aplicativo terá acesso concedido a ele.

dwPreferredProtocols

Fornece uma máscara de bits de protocolos aceitáveis para a conexão. Os valores possíveis, que podem ser combinados com as operações OR são:

Valor	Significado
SCARD_PROTOCOL_T0	T=0 é um protocolo aceitável.
SCARD_PROTOCOL_T1	T=1 é um protocolo aceitável.
0	Este parâmetro só poderá ser zero se dwShareMode é ajustado para SCARD_SHARE_DIRECT. Neste caso, não haverá nenhum driver até que uma diretiva de controle IOCTL_SMARTCARD_SET_PROTOCOL seja enviada com CasControl.

phCard

Recebe um identificador que identifica a conexão ao cartão inteligente na leitora designada.

pdwActiveProtocol
Recebe o marcador que indica o protocolo ativo estabelecido. Possíveis valores:

Valor	Significado
SCARD_PROTOCOL_T0	T=0 é o protocolo ativo.
SCARD_PROTOCOL_T1	T=1 é o protocolo ativo.
SCARD_PROTICOL_UNDEFINED	SCARD_SHARE_DIRECT foi especificado, de forma que nenhuma negociação de protocolo tenha ocorrido. É possível que não haja cartão na leitora.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasConnect é uma função de acesso do cartão inteligente e da leitora.

6.3.5) CasControl()

A função CasControl dá a você controle direto sobre a leitora. Você pode selecioná-la a qualquer momento após uma seleção bem sucedida para CasConnect e antes de uma seleção bem sucedida para CasDisconnect. O efeito no status da leitora depende do código de controle.

```
LONG CasControl(
                IN SCARDHANDLE hCard
                IN DWORD dwControlCode,
                IN LPCVOID lplnBuffer,
                IN DWORD nlnBufferSize
                OUT LPVOID lpOutBuffer,
                IN DWORD nOutBufferSize,
                OUT LPDWORD lpBytesReturned
);
```

Parâmetros

hCard
Este é o valor de referência retornado do CasConnect.

dwControlCode
Fornece o código de controle para a operação. Este valor identifica a operação específica a ser realizada.

lplnBuffer
Fornece um apontador para um buffer que contém os dados necessários para realizar a operação. Este parâmetro pode ser NULL se o parâmetro dwControlCode especificar uma operação que não exija dados de entrada.

nlnBufferSize
Especifica o tamanho, em bytes, do buffer apontado pelo lplnBuffer.

lpOutBuffer
Aponta para um buffer que recebe os dados de saída da operação. Este parâmetro pode ser NULL se o parâmetro dwControlCode especificar uma operação que não produza dados de saída.

nOutBufferSize
Especifica o tamanho, em bytes, do buffer apontado pelo lpOutBuffer.

lpBytesReturned
Aponta para um DWORD que recebe o tamanho, em bytes, dos dados armazenados no buffer apontado pelo lpOutBuffer.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasControl é uma função de acesso Direto do cartão

6.3.6) CasDisconnect()

A função CasDisconnect conclui uma conexão previamente aberta entre a aplicativo de seleção e um cartão inteligente na leitora alvo.

```
LONG CasDisconnect(
        IN SCARDHANDLE hCard,
        IN DWORD dwDisposition
);
```

Parâmetros
hCard
Fornece o valor de referência obtido de uma seleção anterior para o CasConnect.

dwDisposition
Indica o que fazer com o cartão na leitora conectada, no fechamento. Possíveis valores:

Valor	Significado
SCARD_LEAVE_CARD	Não faça nada em particular.
SCARD_RESET_CARD	Reinicialize o cartão.
SCARD_UNPOWER_CARD	Energize o cartão.
SCARD_EJECT_CARD	Ejete o cartão.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

Se um aplicativo (que selecionou CasConnect previamente) existir sem selecionar CasDisconnect, o cartão é reinicializado automaticamente.

CasDisconnect é uma função de acesso do cartão inteligente e da leitora.

6.3.7) CasEndTransaction()

A função CasEndTransaction completa uma transação previamente declarada, permitindo que outros aplicativo retomem as interações com o cartão.

```
LONG CasEndTransaction(
        IN SCARDHANDLE hCard,
        IN DWORD dwDisposition
);
```

Parâmetros
hCard
Fornece o valor de referência obtido de uma seleção anterior para o CasConnect. Este valor também seria utilizado em uma seleção anterior para CasBeginTransaction.

dwDisposition
Indica o que fazer com o cartão na leitora conectada, no fechamento. Possíveis valores:

Valor	Significado
SCARD_LEAVE_CARD	Não faça nada em particular.
SCARD_RESET_CARD	Reinicialize o cartão.
SCARD_UNPOWER_CARD	Energize o cartão.
SCARD_EJECT_CARD	Ejete o cartão.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasEndTransaction é uma função de acesso do cartão inteligente e da leitora.

6.3.8) CasEstablishContext()

A função CasEstablishContext estabelece o contexto do gerente de recursos (o escopo) dentro do qual as operações da base de dados são realizadas.

```
LONG CasEstablishContext(
                IN DWORD dwScope,
                IN LPCVOID pvReserved1,
                IN LPCVOID pvReserved2,
                OUT LPSCARDCONTEXT phContext
);
```

Parâmetros

dwScope
Fornece o escopo do contexto do gerente de recursos. Possíveis valores:

Valor	Significado
SCARD_SCOPE_USER	As operações da base de dados são realizadas dentro do domínio do usuário.
SCARD_SCOPE_SYSTEM	As operações da base de dados são realizadas dentro do domínio do usuário. (O aplicativo de seleção deve ter permissões de acesso apropriados para qualquer ação da base de dados.)

pvReserved1
Reservado para uso futuro, e deve ser NULL. Reservado para permitir que um aplicativo de gerenciamento privilegiado adequado aja em nome de outro usuário.

pvReserved2
Reservado para uso futuro, e deve ser NULL. Reservado para permitir que um aplicativo de gerenciamento privilegiado adequado aja em nome de outro terminal.

phContext
Recebe um identificador do contexto do gerente de recursos estabelecido. Este identificador pode agora ser fornecido para outras funções que tentem trabalhar dentro deste contexto.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

O identificador de contexto retornado pelo CasEstablishContext pode ser utilizado por consulta à base de dados e funções de gerenciamento.
Para liberar um contexto estabelecido do gerente de recursos, utilize CasReleaseContext.

6.3.9) CasForgetCardType()

A função CasForgetCardType remove o cartão inteligente introduzido do subsistema do cartão inteligente.

```
LONG CasForgetCardType(
                IN SCARDCONTEXT hContext,
                IN LPCTSTR szCardName
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext.

szCardName
Fornece o nome apropriado do cartão a ser removido da base de dados do cartão inteligente.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasForgetCardType é uma função de gerenciamento da base de dados.

6.3.10) CasForgetReader()

A função CasForgetReader remove uma leitora introduzida previamente do controle pelo subsistema do cartão inteligente. Ela é removida da base de dados do cartão inteligente, incluindo de qualquer grupo de leitoras que pode ter sido adicionado a ela.

```
LONG CasForgetReader(
        IN SCARDCONTEXT hContext,
        IN LPCTSTR szReaderName
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext.

szReaderName
Fornece o nome apropriado da leitora a ser removida da base de dados do cartão inteligente.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

Se a leitora especificada é o último membro de um grupo de leitoras, o grupo de leitoras é também removido automaticamente.

CasForgetReader é uma função de gerenciamento da base de dados.

6.3.11) CasForgetReaderGroup()

A função CasForgetReaderGroup remove, do subsistema do cartão inteligente, um grupo de leitoras de cartão inteligente introduzido previamente. Embora esta função libere todas as leitoras de um grupo automaticamente, ela não afeta a existência de leitoras individuais na leitora.

Parâmetros

hContext
Fornece o identificador que identifica o contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext.

szGroupName
Fornece o nome apropriado do grupo de leitoras a ser removido. Grupos de leitoras definidos pelo sistema não podem ser removidos da base de dados.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasForgetReaderGroup é uma função de gerenciamento da base de dados.

6.3.12) CasFreeMemory()

A função CasFreeMemory libera memória que foi retornada do gerente de recursos utilizando o designador de comprimento SCARD_AUTOALLOCATE.

```
LONG CasFreeMemory(
        IN SCARDCONTEXT hContext,
        IN LPVOID pvMem
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos retornado do CasEstablishContext, ou NULL se a função de criação for também especificada como NULL para seu hContext.

pvMem
Fornece o bloco de memória a ser liberado.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

6.3.13) CasGetAttrib()

A função CasSetAttrib obtém os atributos de leitora para o identificador dado. Ela não afeta o status da leitora, do driver ou do cartão.

```
LONG CasGetAttrib(
        IN SCARDHANDLE hCard,
        IN DWORD dwScope,
        OUT LPBYTE pbAttr,
        IN OUT LPDWORD pcbAttrLen
);
```

Parâmetros

hCard
Fornece o valor de referência retornado do CasConnect.

dwAttrib
Fornece o identificador para estabelecer o atributo.

PbAttr
Aponta para um buffer que fornece o atributo cujo ID é fornecido em dwAttrld. Se este valor for NULL, CasGetAttrib ignora o comprimento do buffer fornecido em pcbAttrLength, escreve o comprimento do buffer que seria retornado se este parâmetro não tivesse sido NULL para pcbAttrLength, e retorna um código válido.

pcbAttrLen
Fornece o comprimento do buffer pbAttr em bytes, e recebe o comprimento real do atributo recebido.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasGetAttrib funções de acesso direto do cartão.

6.3.14) CasGetCardTypeProviderName()

A função CasGetCardTypeProviderName retorna o nome do módulo (biblioteca de ligação dinâmica) contendo o fornecedor para um nome de cartão dado e tipo de fornecedor.

```
LONG CasGetCardTypeProviderName(
                IN SCARDCONTEXT hContext,
                IN LPCTSTR szCardName,
                IN DWORD dwProviderId,
                OUT LPTSTR szProvider,
                IN OUT LPDWORD pcchProvider
);
```

Parâmetros

hContext
Fornece o identificador que identifica o contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext. Este valor pode ser NULL se a seleção para CasGetCardTypeProviderName não for direcionado para um contexto específico.

szCardName
Fornece o nome do tipo de cartão com o qual o nome do fornecedor é associado.

dwProviderId
Fornece o identificador para o fornecedor associado com este tipo de cartão. O valor fornecido para dwProviderId poderá ser um dos seguintes.

Valor	Ação
SCARD_PROVIDER_PRIMARY	A função recupera o nome do fornecedor primário do serviço de cartão inteligente como uma sequência GUID
SCARD_PROVIDER_CSP	A função recupera o nome do fornecedor do serviço criptográfico

szProvider
Variável de sequência que conterá o nome do fornecedor recuperado após a conclusão bem sucedida desta função.

pcchProvider
Apontador para o valor DWORD. Na entrada, *pcchProvider fornece o comprimento do buffer szProvider em caracteres. Se este valor for SCARD_AUTOALLOCATE, então o szProvider é convertido de um apontador para um apontador em sequência e recebe o endereço do bloco de memória contendo a sequência. Este bloco de memória deve ser desalojada selecionando CasFreeMemory. Na saída, este valor representa o número real de caracteres, incluindo o caractere nulo, na variável szProvider.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

Após a conclusão bem sucedida desta função, o valor no szProvider pode ser utilizado com terceiro parâmetro em uma seleção para CryptAcquireContext.

Exemplo de Código

```
  CASCONTEXT hContext = NULL;

  LPTSTR szProvider = NULL;
  LPTSTR szCardName = _T("WindowsCard");

  DWORD chProvider = SCARD_AUTO_ALLOCATE;

  LONG IReturn = SCARD_S_SUCCESS;

  // Estabeleça um contexto de recursos de cartão inteligente.
            IReturn = CasEstablishContext(SCARD_SCOPE_USER, NULL, NULL, &hContext);

  If (SCARD_S_SUCCESS == IReturn)
  {
  // Recupere o nome do fornecedor.
            IReturn = CasGetCardTypeProviderName(hContext, szCardName, SCARD_PROVIDER_CSP, (LPSTR)&szProvider,
&chProvider);
  }

  If (SCARD_S_SUCCESS == IReturn)
  {
            BOOL fSts = TRUE;
            HCRYPTPROV hProv = NULL;

            // Adquira um contexto de operação criptográfica.
            fSts = CryptAcquireContext(&hProv, NULL, szProvider, PROV_RSA_FULL, 0);

            // Realize operações criptográficas com cartão inteligente…

            // Libere memória alocada por CasGetCardTypeProviderName
            IReturn = CasFreeMemory(hContext, szProvider);
  }
```

6.3.15) CasGetProviderId()

A função CasGetProviderId retorna o identificador (GUID) do fornecedor do serviço primário para o cartão dado.

O selecionador fornece o nome de um cartão inteligente (previamente introduzido para o sistema) e recebe o identificador registrado do fornecedor GUID do serviço primário, se houver um.

```
LONG CasGetProviderId(
                IN SCARDCANTEXT hContext,
                IN LPCTSTR szCard,
                OUT LPGUID pguidProviderId
);
```

Parâmetros
hContext
Fornece o identificador do contexto do gerente de recursos para a consulta. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext, ou ajusta NULL se a consulta não é direcionada para um contexto específico.

szCard
Fornece o nome do cartão definido para o sistema.

pguidProviderId
Recebe o identificador (GUID) do fornecedor do serviço primário. Seu fornecedor pode ser via COM, e fornecerá acesso a outros serviços no cartão.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasGetProviderId é uma função de consulta à base de dados.

6.3.16) CasGetStatusChange()

A função CasGetStatusChange bloqueia a execução até que a disponibilidade atual de cartões em um conjunto específico de leitoras mude.

O selecionador fornece uma lista de leitoras que devem ser monitoradas por uma matriz SCARD_READERSTATE e a quantidade máxima de tempo (em milissegundos) que se deve aguardar para que uma ação ocorra em uma das leitoras listadas. Note que CasGetStatusChange utiliza o valor fornecido pelo usuário nos membros dwCurrentState do parâmetro rgReaderStates, como a definição do status atual das leitoras. A função retorna quando há uma mudança de disponibilidade, tendo preenchido nos membros dwEventState do parâmetro rgReaderStates apropriadamente.

```
LONG casGetStatusChange(
                IN SCARDCONTEXT hContext,
                IN DWORD dwTimeout,
                IN OUT LPSCARD_READERSTATE rgReaderStates,
                IN DWORD cReaders
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext.

dwTimeout
Fornece a quantidade máxima de tempo (em milissegundos) para aguardar uma ação. Um valor zero implica em um valor de INFINITE, dwTimeout nunca se esgotará.

rgReaderStates
Fornece uma matriz de estruturas SCARD_READERSTATE que especifica quais as leitoras a serem observadas, e recebe o resultado.

cReaders
Fornece o número de elementos na matriz rgReaderStates.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasGetStatusChange é uma função de rastreabilidade do cartão inteligente.

6.3.17) CasIntroduceCardType()

A função CasIntroduceCardType apresenta um cartão inteligente ao subsistema do cartão inteligente (para um usuário ativo) adicionando-o à base de dados do cartão inteligente.

```
LONG CasIntroduceCardType(
                IN SCARDCONTEXT hContext,
                IN LPCTSTR szCardName,
                IN LPGUID pguidPrimaryProvider,
                IN LPGUID rgguidInterfaces,
                IN DWORD dwInterfaceCount,
                IN LPCBYTE pbAtr,
                IN LPCBYTE pbAtrMask,
                IN DWORD cbAtrLen
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext.

szCardName
Fornece o nome pelo qual o usuário pode reconhecer o cartão.

pguidPrimaryProvider
Aponta para o identificador (GUID) para o fornecedor do serviço primário para o cartão inteligente.

rgguidInterfaces
Fornece uma matriz de identificadores (GUIDs) que identificam as interfaces suportadas pelo cartão inteligente.

dwInterfaceCount
Fornece o número de identificadores na matriz rgguidInterfaces.

pbAtr
Fornece uma sequência de ATR que pode ser utilizada para fins de correspondência ao consultar a base de dados do cartão inteligente (vide CasListCards). O comprimento da sequência é determinado por análise normal de ATR.

pbAtrMask
Fornece uma máscara de bits opcional para ser utilizada ao compara os ATRs dos cartões inteligentes para o ATR fornecido em pbAtr. Se este valor é não-NULL, ele deve apontar para uma sequência de bytes com o mesmo comprimento da sequência de ATR fornecida em pbAtr. Quando uma dada sequência de ATR 'A' é comparada com o ATR fornecido em pbAtr, ela corresponde se e apenas se A & M = pbAtr, onde M é a máscara fornecida, e & representa o AND lógico bit a bit.

cbAtrLen
Fornece o comprimento do ATR e a Máscara de ATR opcional. Se este valor for zero, então o comprimento do ATR é determinado por análise normal de ATR. Este valor não pode ser zero ser um valor de pbAtrMask fornecido.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações
CasIntroduceCardType é uma função de gerenciamento da base de dados. Para remover um cartão inteligente, utilize CasForgetCardType.

6.3.18) CasIntroduceReader()

A função CasIntroduceReader apresenta um novo nome para uma leitora de cartão inteligente existente.

Nota: leitoras de cartões inteligentes são automaticamente apresentados ao sistema; um programa de configuração de leitora de cartão inteligente também pode apresentar uma leitora de cartão inteligente para o sistema.

```
LONG CasIntroduceReader(
                IN SCARDCONTEXT hContext,
                IN LPCTSTR szReaderName,
                IN LPCTSTR szDeviceName
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext.

szReaderName
Fornece o nome apropriado a ser atribuído para a leitora.

szDeviceName
Fornece o nome do sistema da leitora do cartão inteligente.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

Todas as leitoras instaladas no sistema são automaticamente introduzidas por seus nomes de sistema. Normalmente, CasIntroduceReader é selecionado apenas para mudar o nome de uma leitora existente.

CasIntroduceReader é uma função de gerenciamento da base de dados. Para uma descrição de outras funções de gerenciamento de base de dados, consulte as Funções de Gerenciamento da Base de Dados.

Para remover uma leitora, utilize CasForgetReader.

6.3.19) CasIntroduceReaderGroup()

A função CasIntroduceReaderGroup apresenta um grupo de leitoras para o subsistema do cartão inteligente. No entanto, o grupo de leitoras não é criado até que o grupo seja especificado ao adicionar uma leitora à base de dados do cartão inteligente.

```
LONG CasIntroduceReaderGroup(
                IN SCARDCONTEXT hContext,
                IN LPCTSTR szGroupName
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext.

szGroupName
Fornece o nome apropriado a ser atribuído para o novo grupo de leitoras.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasIntroduceReaderGroup é fornecido para compatibilidade de especificação PC/SC. Grupos de leitoras não serão armazenados até que uma leitora seja adicionada ao grupo.

CasIntroduceReaderGroup é uma função de gerenciamento da base de dados.

Para remover um grupo de leitoras, utilize CasForgetReaderGroup.

6.3.20) CasListCards()

A função CasListCards busca a base de dados do cartão inteligente e fornece uma lista de cartões designados apresentados previamente ao sistema pelo usuário.

O selecionador especifica uma sequência ATR, um conjunto de identificadores de interface (GUIDs), ou ambos. Se tanto uma sequência ATR quanto uma matriz de identificador forem fornecidos, os cartões retornados corresponderão à sequência ATR fornecida e suportarão as interfaces especificadas.

```
LONG CasListCards(
        IN SCARDCONTEXT hContext,
        IN LPCBYTE pbAtr,
        IN LPCGUID rgguidInterfaces,
        IN DWORD cguidInterfaceCount,
        OUT LPTSTR mszCards,
        IN OUT LPDWORD pcchCards
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos para a consulta. O contexto do gerente de recursos pode ser ajustado por uma seleção prévia por CasEstablishContext, ou ajusta NULL se a consulta não é direcionada para um contexto específico.

pbAtr
Fornece o endereço de uma sequência ATR para comparar com cartões conhecidos, ou NULL se não for realizada nenhuma correspondência de ATR.

rgguidInterfaces
Fornece uma matriz de identificadores (GUIDs) ou NULL se nenhuma correspondência de interface for realizada. Quando uma matriz é fornecida, um nome de cartão será retornado apenas se todos os identificadores especificados forem suportados pelo cartão.

cguidInterfaceCount
Fornece o número de entradas na matriz rgguidInterfaces. Se rgguidInterfaces for NULL, então este valor será ignorado.

mszCards
Recebe uma multi sequência que lista os cartões inteligentes encontrados. Se este valor for NULL, CasListCards ignora o comprimento do buffer fornecido em pcchCards, retornando o comprimento do buffer que seria retornado se este parâmetro não tivesse sido NULL para pcchCards, e retorna um código válido.

pcchCards
Fornece o comprimento do buffer mszCards em caracteres, e recebe o comprimento real da estrutura multi sequência, incluindo todos os caracteres nulos à direita. Se o comprimento de buffer for especificado como SCARD_AUTOALLOCATE, então o szCards é convertido de um apontador para um apontador em sequência e recebe o endereço do bloco de memória contendo a multi sequência. Este bloco de memória deve ser desalojado selecionando CasFreeMemory.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações
Para retornar todos os cartões inteligentes apresentados ao subsistema, ajuste pbAtr e rgguidInterfaces para NULL. CasListCards é uma função de consulta à base de dados.

6.3.21) CasListInterfaces()

A função CasListInterfaces fornece uma lista de interfaces fornecidas por um dado cartão.

O selecionador fornece o nome de um cartão inteligente previamente apresentado para o subsistema, e recebe a lista de interfaces suportadas pelo cartão.

```
LONG CasListInterfaces(
                IN SCARDCONTEXT hContext,
                IN LPCTSTR szCard,
                OUT LPGUID pguidInterfaces,
                IN OUT LPDWORD pcguidInterfaces
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos para a consulta. O contexto do gerente de recursos pode ser ajustado por uma seleção prévia para CasEstablishContext, ou ajusta NULL se a consulta não é direcionada para um contexto específico.

szCard
Fornece o nome do cartão inteligente já apresentado para o subsistema do cartão inteligente.

pguidInterfaces
Fornece uma matriz de identificadores de interface (GUIDs) que indicam as interfaces suportadas pelo cartão inteligente. Se este valor for NULL, CasListInterfaces ignora o comprimento da matriz fornecido em pcguidInterfaces, retornando o tamanho da matriz que seria retornado se este parâmetro não tivesse sido NULL para pcguidInterfaces, e retorna um código válido.

pcguidInterfaces
Fornece o tamanho da matriz pguidInterfaces, e recebe o tamanho real da matriz retornada. Se o tamanho da matriz for especificado como SCARD_AUTOALLOCATE, então o pcguidInterfaces é convertido de um apontador para um apontador GUID, e recebe o endereço do bloco de memória contendo a matriz. Este bloco de memória deve ser desalojado selecionando CasFreeMemory.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasListInterfaces é uma função de consulta à base de dados.

6.3.22) CasListReaderGroups()

A função CasListReaderGroups fornece uma lista de grupos de leitoras que foram apresentados previamente para o sistema.

```
LONG CasListReaderGroups(
                IN SCARDCONTEXT hContext,
                OUT LPTSTR mszGroups,
                IN OUT LPDWORD pcchGroups
);
```

Parâmetros

hContext
Fornece o identificador que identifica o contexto do gerente de recursos para a consulta. O contexto do gerente de recursos pode ser ajustado por uma seleção prévia para CasEstablishContext, ou ajusta NULL se a consulta não é direcionada para um contexto específico.

mszGroups
Recebe uma multi sequência que lista os grupos de leitoras definidos para o sistema e disponíveis para o usuário atual no terminal atual. Se este valor for NULL, CasListReaderGroups ignora o comprimento do buffer fornecido em pcchGroups, escreve o comprimento do buffer que seria retornado se este parâmetro não tivesse sido NULL para pcchGroups, e retorna um código válido.

pcchGroups
Fornece o comprimento do buffer mszGroups em caracteres, e recebe o comprimento real da estrutura multi sequência, incluindo todos os caracteres nulos à direita. Se o comprimento de buffer for especificado como SCARD_AUTOALLOCATE, então o szGroups é convertido de um apontador para um apontador em sequência, e recebe o endereço do bloco de memória contendo a multi sequência. Este bloco de memória deve ser desalojado selecionando CasFreeMemory.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

Um grupo é retornado somente se conter pelo menos uma leitora. Isto inclui o grupo Scard$DefaultReaders. O grupo Scard$AllReaders não pode ser retornado, já que existe apenas implicitamente.

CasListReaderGroups é uma função de consulta à base de dados.

6.3.23) CasListReaders()

A função CasListReaders fornece uma lista de leitoras dentro de um conjunto de grupos de leitoras designadas, eliminando duplicatas.

O selecionador fornece um grupo de leitoras, e recebe a lista de leitoras dentro dos grupos designados. Nomes de grupos não reconhecidos serão ignorados.

```
LONG CasListReaders(
        IN SCARDCONTEXT hContext,
        IN LPCTSTR mszGroups,
        OUT LPTSTR mszReaders,
        IN OUT LPDWORD pcchReaders
);
```

Parâmetros

hContext
Fornece o identificador que identifica o contexto do gerente de recursos para a consulta. O contexto do gerente de recursos pode ser ajustado por uma seleção prévia para CasEstablishContext, ou ajusta NULL se a consulta não é direcionada para um contexto específico.

mszGroups
Fornece o nome dos grupos de leitoras definidos para o sistema, como uma multi sequência. Utilize um valor NULL para listar todas as leitoras no sistema (isto é, o grupo SCard$AllReaders).

mszReaders
Recebe uma multi sequência que lista as leitoras de cartões dentro dos grupos de leitoras fornecidos. Se este valor for NULL, CasListReaders ignora o comprimento do buffer fornecido em pcchReaders, escreve o comprimento do buffer que seria retornado se este parâmetro não tivesse sido NULL para pcchReaders, e retorna um código válido.

pcchReaders
Fornece o comprimento do buffer mszReaders em caracteres, e recebe o comprimento real da estrutura multi sequência, incluindo todos os caracteres nulos à direita. Se o comprimento de buffer for especificado como SCARD_AUTOALLOCATE, então o szReaders é convertido de um apontador para um apontador em sequência, e recebe o endereço do bloco de memória contendo a estrutura de multi sequência. Este bloco de memória deve ser desalojado selecionando CasFreeMemory.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasListReaders é uma função de consulta à base de dados.

6.3.24) CasLocateCards()

A função CasLocateCards busca as leitoras listadas no parâmetro rgReaderStates para um cartão com uma sequência ATR que corresponde a um dos nomes de cartão especificados no mszCards, retornando imediatamente com o resultado.

```
LONG CasLocateCards(
                IN SCARDCONTEXT hContext,
                IN LPCTSTR mszCards,
                IN OUT LPSCARD_READERSTATE rgReaderStates,
                IN DWORD cReaders
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext.

mszCards
Fornece uma multi sequência que contém os nomes dos cartões a serem buscados.

rgReaderStates
Fornece uma matriz de estruturas SCARD_READERSTATE que especifica as leitoras a serem buscadas, e recebe o resultado.

cReaders
Fornece o número de elementos na matriz rgReaderStates.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

Este serviço é especialmente útil quando usado em conjunto com CasGetStatusChange. Se nenhum cartão correspondente é encontrado por meio de CasLocateCards, o aplicativo de seleção poderá utilizar CasGetStatusChange para aguardar por mudanças na disponibilidade do cartão.

CasLocateCards é uma função de rastreabilidade do cartão inteligente.

6.3.25) CasReconnect()

A função CasReconnect restabelece uma conexão existente entre o aplicativo de seleção e um cartão inteligente. Esta função muda um identificador do cartão de acesso direto para acesso geral, ou reconhece e limpa uma condição de erro que está impedindo outros acessos ao cartão.

```
LONG CasReconnect(
        IN SCARDHANDLE hCard,
        IN DWORD dwShareMode,
        IN DWORD dwPreferredProtocols,
        IN DWORD dwInitialization,
        OUT LPDWORD pdwActiveProtocol
);
```

Parâmetros

hCard
Fornece o valor de referência obtido de uma seleção anterior para o CasConnect.

dwShareMode
Fornece um marcador que indica se outros aplicativos podem formar conexões com este cartão. Possíveis valores:

Valor	Significado
SCARD_SHARE_SHARED	Este aplicativo compartilhará o cartão com outros aplicativos.
SCARD_SHARE_EXCLUSIVE	Este aplicativo não compartilhará o cartão com outros aplicativos.

dwPreferredProtocols
Fornece uma máscara de bits de protocolos aceitáveis para esta conexão. Os valores possíveis, que podem ser combinados com a operação OR, são:

Valor	Significado
SCARD_PROTOCOL_T0	T=0 é um protocolo aceitável.
SCARD_PROTOCOL_T1	T=1 é um protocolo aceitável.

dwInitialization
Indica que tipo de inicialização deve ser realizado no cartão. Possíveis valores:

Valor	Significado
SCARD_LEAVE_CARD	Não faça nada em particular na reconexão.
SCARD_RESET_CARD	Reinicialize o cartão (Warm Reset)
SCARD_UNPOWER_CARD	Desconecte o cartão e reinicialize-o (Cold Reset)

pdwActiveProtocol
Recebe o marcador que indica o protocolo ativo estabelecido. Possíveis valores:

Valor	Significado
SCARD_PROTOCOL_T0	T=0 é o protocolo ativo.
SCARD_PROTOCOL_T1	T=1 é o protocolo ativo.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasReconnect é uma função de acesso do cartão inteligente e da leitora.

6.3.26) CasReleaseContext()

A função CasReleaseContext fecha um contexto do gerente de recursos estabelecido, liberando qualquer recurso alocado sob aquele contexto, incluindo objetos SCARDHANDLE e memória alocada utilizando o designador de comprimento SCARD_AUTOALLOCATE.

```
LONG CasReleaseContext(
        IN SCARDCONTEXT hContext
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

6.3.27) CasRemoveReaderFromGroup()

A função CasRemoveReaderFromGroup remove uma leitora de um grupo de leitoras existente. Esta função não afeta a leitora ou grupo de leitoras.

```
LONG CasRemoveReaderFromGroup(
        IN SCARDCONTEXT hContext,
        IN LPCTSTR szReaderName,
        IN LPCTSTR szGroupName
);
```

Parâmetros

hContext
Fornece o identificador do contexto do gerente de recursos. O contexto do gerente de recursos é ajustado por uma seleção prévia para CasEstablishContext.

szReaderName
Fornece o nome apropriado da leitora a ser removida.

szGroupName
Fornece o nome apropriado do grupo do qual a leitora deve ser removida.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

Quando a última leitora for removida de um grupo, o grupo é automaticamente esquecido.

CasRemoveReaderFromGroup é uma função de gerenciamento da base de dados.
Para adicionar uma leitora ao grupo de leitoras, utilize CasAddReaderToGroup.

6.3.28) CasSetAttrib()

A função CasSetAttrib estabelece o atributo da leitora dado para o manipulador dado. Ela não afeta o status da leitora, do driver da leitora ou do cartão inteligente. Nem todos os atributos são suportados por todas as leitoras (nem podem eles ser ajustados a todo o momento), pois muitos dos atributos estão sob controle direto do protocolo de transporte.

```
LONG CasSetAttrib(
                IN SCARDHANDLE hCard,
                IN dwAttrId,
                IN LPCBYTE pbAttr,
                IN DWORD cbAttrLen
);
```

Parâmetros

hCard
Fornece o valor de referência retornado do CasConnect.

dwAttrId
Fornece o identificador para estabelecer o atributo.

PbAttr
Aponta para um buffer que fornece o atributo cujo ID é fornecido em dwAttrId.

cbAttrLen
Fornece o comprimento (em bytes) do valor do atributo no buffer pbAttr.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasSetAttrib é uma função de acesso direto do cartão.

6.3.29) CasStatus()

A função CasStatus fornece o status atual de um cartão inteligente em uma leitora. Você pode selecioná-la a qualquer momento após uma seleção bem sucedida para CasConnect e antes de uma seleção bem sucedida para CasDisconnect. Ela não afeta o status da leitora, ou do driver da leitora.

```
LONG CasStatus(
        IN SCARDHANDLE hCard,
        OUT LPTSTR szReaderName,
        IN OUT LPDWORD pcchReaderLen,
        OUT LPDWORD pdwState,
        OUT LPDWORD pdwProtocol,
        OUT LPBYTE pbAtr,
        OUT LPDWORD pcbAtrLen
);
```

Parâmetros

hCard
Este é o valor de referência retornado do CasConnect.

szReaderName
Recebe uma lista de nomes adequados (multi sequência) pelo qual a leitora conectada atualmente é conhecida.

pcchReaderLen
Na entrada, fornece o comprimento do buffer szReaderName
Na saída, recebe o comprimento real (em caracteres) da lista de nomes das leitoras, incluindo o caractere NULL à direita.

pdwState
Recebe o status atual do cartão inteligente na leitora. Após a validação, ela recebe um dos indicadores de status a seguir:

Valor	Significado
SCARD_ABSENT	Não há cartão na leitora.
SCARD_PRESENT	Há um cartão na leitora, mas ele não foi colocado na posição de uso.
SCARD_SWALLOWED	Há um cartão na leitora em posição de uso. O cartão não está energizado.
SCARD_POWERED	Há energia sendo fornecida para o cartão, mas o driver da leitora não conhece o modo do cartão.
SCARD_NEGOTIABLE	O cartão foi reinicializado, e está aguardando a negociação PTS.
SCARD-SPECIFIC	O cartão foi reinicializado, e foram estabelecidos protocolos específicos de comunicação.

pdwProtocol
Recebe o protocolo atual, se houver. O valor retornado só faz sentido se o valor retornado de pdwState for SCARD_SPECIFICMODE. Os possíveis valores retornados são os seguintes:

Valor	Significado
SCARD_PROTOCOL_RAW	O protocolo de transferência RAW está em uso.
SCARD_PROTOCOL_T0	O protocolo ISO 7816/3 T=0 está em uso.
SCARD_PROTOCOL_T1	O protocolo ISO 7816/3 T=1 está em uso.

pbAtr
Aponta para um buffer de 32 bytes que recebe a sequência ATR do cartão inserido atualmente, se disponível.

pcbAtrLen
Aponta para um DWORD para receber o número de bytes na sequência ATR (32 bytes, no máximo).

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

Observações

CasStatus é uma função de acesso do cartão inteligente e da leitora.

6.3.30) CasTransmit()

A função CasTransmit envia uma solicitação de serviço para o cartão inteligente, e espera receber dados de volta do cartão.

```
LONG CasTransmit(
        IN SCARDHANDLE hCard,
        IN LPCSCARD_I0_REQUEST pioSendPci,
        IN LPCBYTE pbSendBuffer,
        IN DWORD cbSendLength,
        IN OUT LPSCARD_IO_REQUEST pioRecvPci,
        OUT LPBYTE pbRecvBuffer,
        IN OUT LPDWORD pcbRecvLength
);
```

Parâmetros

hCard
Fornece o valor de referência retornado do CasConnect.

pioSendPci
Aponta para a estrutura de cabeçalho do protocolo para a instrução. Este buffer está no formato de uma estrutura SCARD_IO_REQUEST, seguida da informação de controle do protocolo (PCI).
Para protocolos T=0, T=1, e RAW, a estrutura PCI é constante. O subsistema do cartão inteligente fornece uma estrutura global de PCI T=0, T=1, ou RAW, que você pode consultar utilizando os símbolos SCARD_PCI_T0, SCARD_PCI_T1, e SCARD_PCI_RAW respectivamente.

pbSendBuffer
Aponta para os dados reais a serem escritos para o cartão.
T=0 Nota: para T=0, os parâmetros dos dados são colocados no pbSendBuffer de acordo com a seguinte estrutura:

```
struct {
        BYTE
        bCla,  // A classe de instrução
        bIns,  // O código de instrução
        bP1,   // O parâmetro para a instrução
        bP2,   // O parâmetro para a instrução
        bP3;   // Tamanho da Transferência de I/O
    } CmdBytes;
```

Membros:

bCla
A classe de instrução T=0

bIns
Um código de instrução na classe de instrução T=0

bP1, bP2
Códigos de referência que completam o código de instrução

bP3
O número de bytes de dados que devem ser transmitidos durante o comando, de acordo com a ISO 7816-4, Seção 8.2.1.

Os dados enviados ao cartão devem seguir imediatamente o buffer de envio. No caso especial em que nenhum dado é enviado para o cartão e nenhum dado é esperado de volta, bP3 não é enviado.

cbSendLength
Fornece o comprimento (em bytes) do parâmetro pbSendBuffer.
T=0 Nota: para T=0, no caso especial em que nenhum dado é enviado para o cartão e nenhum dado é esperado de volta, este comprimento deve refletir que o membro bP3 não está sendo enviado: o comprimento deve ser sizeof(CmdBytes) – sizeof(BYTE).

pioRecvPci
Aponta para a estrutura de cabeçalho do protocolo para a instrução, seguido por um buffer para receber qualquer informação de controle de protocolo (PCI) retornado específico do protocolo em uso. Este parâmetro pode ser NULL se não houver nenhum PCI retornado desejável.

pbRecvBuffer
Aponta para qualquer dado retornado do cartão.
T=0 Nota: para T=0, os dados são imediatamente seguidos pelos bytes de status SW1 e SW2. Se nenhum dado for retornado do cartão, então este buffer conterá apenas os bytes de status SW1 e SW2.

pcbRecvLength
Fornece o comprimento do parâmetro pbRecvBuffer (em bytes) e recebe o número real de bytes recebidos do cartão inteligente.
T=0 Nota: para T=0, o buffer de recebimento deve ter o comprimento de pelo menos dois bytes, a fim de receber os bytes de status SW1 e SW2.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar **Códigos de Erro**).

Observações

CasTransmit é uma função de acesso do cartão inteligente e da leitora.

Observações do protocolo T=0

Para o protocolo T=0, os dados recebidos de volta são os códigos de status SW1 e SW2, possivelmente precedidos de dados de resposta. Os parágrafos a seguir fornecem informações nos buffers de envio e recebimento usados para transferir dados e emitir um comando.

Envio de Dados para o Cartão

Para enviar n bytes de dados para o cartão, onde n>0, os buffers de envio e recebimento devem ser formatados como a seguir.

Os primeiros quatro bytes do buffer pbSendBuffer contém os valores de CLA, INS, P1, e P2 para a operação T=0. O quinto byte deve ser ajustado para n: o tamanho (em bytes) dos dados a serem transferidos para o cartão. Os próximos n bytes deverão conter os dados a serem enviados para o cartão.

O parâmetro cbSendLength deverá ser ajustado para o tamanho da informação de cabeçalho T=0 (CLA, INS, P1 e P2), mais um byte contendo o comprimento dos dados a serem transferidos (n), mais o tamanho dos dados a serem enviados. Neste exemplo, este é n+5.

O pbRecvBuffer receberá os códigos de status SW1 e SW2 da operação.

O pcbRecvLength deve ser pelo menos 2, e será ajustado para 2 depois do retorno.

Obtenção de Dados do Cartão

Para receber n>0 bytes de dados do cartão, os buffers de envio e recebimento devem ser formatados como a seguir.

Os primeiros quatro bytes do buffer pbSendBuffer contém os valores de CLA, INS, P1, e P2 para a operação T=0. O quinto byte deve ser ajustado para n: o tamanho (em bytes) dos dados a serem transferidos para o cartão. Se 256 bytes devem ser transferidos para o cartão, então este byte deve ser ajustado para zero.

O parâmetro cbSendLength deverá ser ajustado para 5, o tamanho da informação do cabeçalho de T=0.

O pbRecvBuffer receberá os dados retornados do cartão, imediatamente seguidos pelos códigos de status SW1 e SW2 da operação.

O pcbRecvLength deve ser pelo menos n+2, e será ajustado para n+2 depois do retorno.

Emissão de um Comando sem Troca de Dados

Para emitir um comando para o cartão que não envolve a troca de dados (enviados ou recebidos), os buffers de envio e recebimento devem ser formatados como a seguir.

O buffer pbSendBuffer deverá conter os valores de CLA, INS, P1, e P2 para a operação T=0. O valor de P3 não é enviado. (Isto é para diferenciar o cabeçalho do caso em que se espera que 256 bytes sejam retornados.)

O parâmetro cbSendLength deverá ser ajustado para 4, o tamanho da informação do cabeçalho de T=0 (CLA, INS, P1, e P2).

O pbRecvBuffer receberá os códigos de status SW1 e SW2 da operação.

O pcbRecvLength deve ser pelo menos 2, e será ajustado para 2 depois do retorno.

6.3.31) CasUIDlgSelectCard()

A função CasUIDlgSelectCard exibe o diálogo "select card" do cartão inteligente.

```
LONG (
                                        IN LPOPENCARDNAME_EX pDlgStruc
);
```

Parâmetros

pDlgStruc
Aponta para a estrutura OPENCARDNAME_EX para o diálogo "select card".

Observações

A função CasUIDlgSelectCard fornece um método para conectar a um cartão inteligente específico. Ao ser selecionada, esta função realiza uma busca por cartões inteligentes adequados que correspondam ao membro OPENCARD_SEARCH_CRITERIA de *pDlgStruc. Dependendo do membro dwFlags do *pDlgStruc, esta função assume as seguintes ações.

Valor para dwFlag	Ação
C_DLG_FORCE_UI	Se conecta ao cartão selecionado pelo usuário do diálogo "select card" do cartão inteligente.
SC_DLG_MINIMAL_UI	Seleciona o cartão inteligente se apenas um se enquadra ao critério, ou retorna informação sobre a seleção do usuário se mais de um cartão inteligente se enquadra ao critério.
SC_DLG_NO_UI	Seleciona o primeiro cartão disponível.

Valores de Retorno

Se a função for...	O valor de retorno será…
Válido	SCARD_S_SUCCESS
Inválido	Um código de erro aparecerá (para uma lista de códigos de erro, consultar Códigos de Erro).

6.4) Códigos de Erros do Cartão Inteligente

Esta seção descreve os códigos de erro primário retornados pelas funções do cartão inteligente.

Erro	Definição
SCARD_E_CANCELLED	A ação foi cancelada por uma solicitação CasCancel.
SCARD_E_CANT_DISPOSE	O sistema não pode descartar a mídia da maneira solicitada.
SCARD_E_CARD_UNSUPPORTED	O cartão inteligente não possui os requisitos mínimos para suporte.
SCARD_E_DUPLICATE_READER	O driver da leitora não produziu um nome de leitora exclusivo.
SCARD_E_INSUFFICIENT_BUFFER	O buffer de dados para dados retornados é muito pequeno para os dados retornados.
SCARD_E_INVALID_ATR	Uma sequência ATR obtida do registro não é uma sequência ATR válida.
SCARD_E_INVALID_HANDLE	O identificador fornecido é inválido.
SCARD_E_INVALID_PARAMETER	Um ou mais dos parâmetros fornecidos não pode ser interpretado adequadamente.
SCARD_E_INVALID_TARGET	Informações de inicialização de registro inexistente ou inválido.
SCARD_E_INVALID_VALUE	Um ou mais dos valores de parâmetros fornecidos não pode ser interpretado adequadamente.
SCARD_E_NOT_READY	A leitora ou o cartão não está pronta(o) para aceitar comandos.
SCARD_E_NOT_TRANSACTED	Houve uma tentativa de terminar uma transação não existente.
SCARD_E_NO_MEMORY	Não há memória suficiente disponível para completar este comando.
SCARD_E_NO_SERVICE	O gerente de recursos do cartão inteligente não está funcionando.
SCARD_E_NO_SMARTCARD	A operação exige um cartão inteligente, mas não há nenhum cartão inteligente no dispositivo no momento.
SCARD_E_PCI_TOO_SMALL	O buffer de recebimento do PCI é muito pequeno.
SCARD_E_PROTO_MISMATCH	Os protocolos solicitados são incompatíveis com o protocolo em uso com o cartão no momento.
SCARD_E_READER_UNAVAILABLE	A leitora especificada não está disponível para uso no momento.
SCARD_E_READER_UNSUPPORTED	O driver da leitora não cumpre os requisitos mínimos para suporte.
SCARD_E_SERVICE_STOPPED	O gerente de recursos do cartão inteligente foi desligado.
SCARD_E_SHARING_VIOLATION	O cartão inteligente não pode ser acessado devido às conexões pendentes.
SCARD_E_SYSTEM_CANCELLED	A ação foi cancelada pelo sistema, provavelmente para se desconectar ou desligar.
SCARD_E_TIMEOUT	O valor do tempo limite especificado pelo usuário expirou.
SCARD_E_UNKNOWN_CARD	O nome do cartão inteligente especificado não foi reconhecido.
SCARD_E_UNKNOWN_READER	O nome da leitora especificada não foi reconhecido.
SCARD_F_COMM_ERROR	Um erro de comunicação interna foi detectado.
SCARD_F_INTERNAL_ERROR	Houve uma falha de consistência interna.
SCARD_F_UNKNOWN_ERROR	Um erro interno foi detectado, mas a fonte é desconhecida.
SCARD_F_WAITED_TOO_LONG	Um temporizador interno de consistência expirou.
SCARD_S_SUCCESS	Nenhum erro foi encontrado.
SCARD_W_REMOVED_CARD	O cartão inteligente foi removido, de forma que não é mais possível haver comunicação.
SCARD_W_RESET_CARD	O cartão inteligente foi reinicializado, então qualquer informação de status compartilhada está inválida.

SCARD_W_UNPOWERED_CARD	A força foi cortada do cartão inteligente, de forma que não é mais possível estabelecer comunicação.
SCARD_W_UNRESPONSIVE_CARD	O cartão inteligente não está respondendo a uma reinicialização.
SCARD_W_UNSUPPORTED_CARD	A leitora não pode se comunicar com o cartão, devido a conflitos de configuração da sequência ATR.

APÊNDICE:

A) Estrutura de Dados TLV ou BER-TLV

O formato BER (Basic Encoding Rules) para ASN.1 é definido pela ISO/IEC 8825.
Estruturas de dados criadas de acordo com estas regras ou formato são chamadas de estruturas de dados TLV ou BER-TLV.
A ISO 7816-4 define o uso de estruturas de dados TLV para troca de dados entre a leitora e o cartão.
A leitora Smartnonus suporta a estrutura de dados TLV ou BER-TLV de acordo com ISO 7816-4.

Na estrutura TLV, T (tag, "etiqueta") define o tipo do objeto, o tipo de dados presente na estrutura, L (length) é o cumprimento ou número de bytes contidos na estrutura, e por fim V (value) são os dados ou valores contidos na estrutura.

T tag	L length	V valve
1 2bytes	1 3bytes	N bytes

Figura A1 - Estrutura de dados TLV

Existe ainda uma variação da BER-TLV identificada como Simple-TLV.
Na estrutura Simple-TLV, é utilizado 1 byte no campo T (tag), e 1-3 bytes no campo L(length), permitindo um valor de 0 a 64k bytes no campo V (value).

B) Suporte Técnico

A Nonus oferece suporte técnico para as leitoras Smartnonus através de telefone ou e-mail:

Tel: (11) 2344-0404 (Opção 4 - Suporte e Assistência Técnica)
e-mail: suporte@nonus.com.br

Antes de entrar em contato com a Nonus tenha em mãos o número de série do equipamento, se possível, cópia da nota fiscal de compra.
Dependendo da natureza do problema, é importante que esteja próximo ao computador onde a leitora esta instalada, e que tenha direitos administrativos sobre o computador para instalação de drivers e/ou diagnósticos.

Para diagnóstico de suporte técnico, o número de série é suficiente para que a Nonus identifique as versões de Hardware e Firmware do equipamento.
As informações sobre o software ou versão de driver instalados, poderão ser obtidas no computador onde a leitora esta instalada.
Abaixo estamos indicando como cada uma destas informações pode ser obtida individualmente.

B.1) Identificação do Hardware

Para obter a versão de hardware da leitora Smartnonus, será preciso abrir o equipamento.
Não realize esta operação a menos que solicitada pelo técnico do suporte Nonus.
Lembramos que a abertura não autorizada do equipamento pode acarretar em perda de garantia.

A versão de hardware pode ser identificada de 2 maneiras:
1) Através de etiqueta adesiva, para as placas sem serigrafia
2) Serigrafia no centro da placa de circuito impresso (figura B.1).

A versão corrente do hardware até a impressão deste manual é: **REV.4**

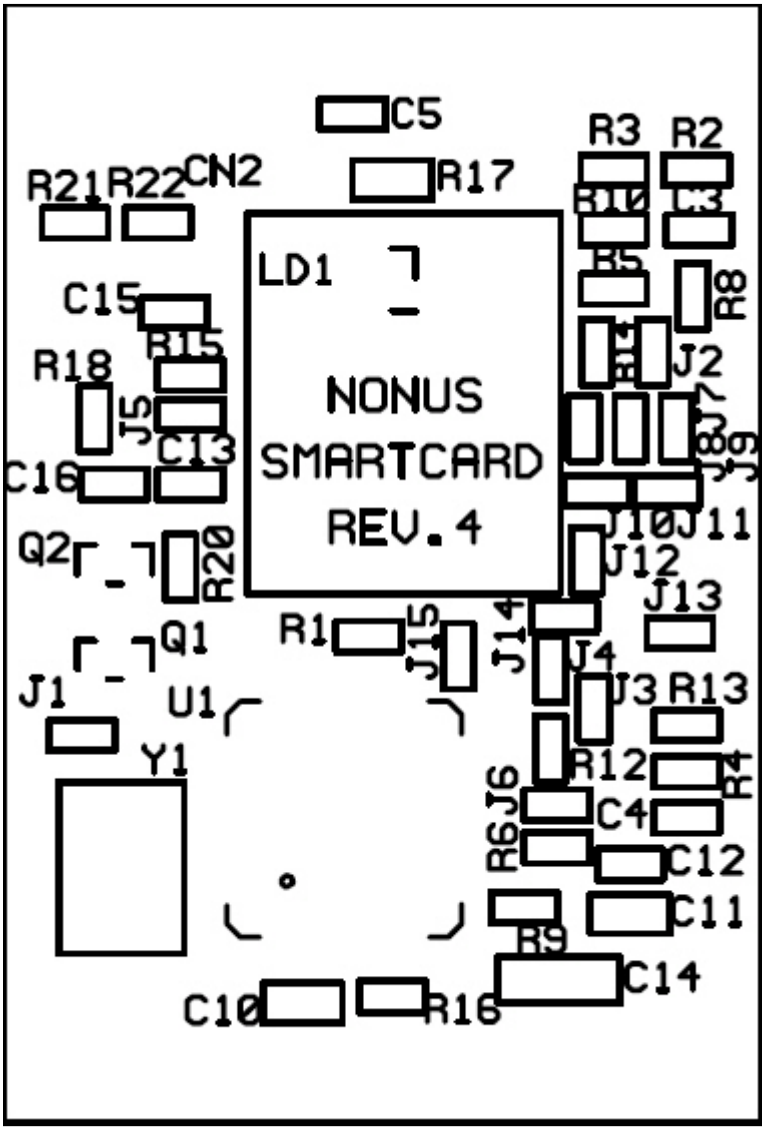


Figura B.1 - Identificação da versão de hardware da leitora Smartnonus.

B.2) Identificação do Firmware.

A versão de firmware da leitora Smartnonus pode ser obtida através do programa de Demonstração da API Smartnonus. Este programa esta disponível no site da Nonus em www.nonus.com.br/smartnonus e pode ser baixado gratuitamente. Além de testar a instalação da leitora, na sua tela principal, permite ler a versão do firmware da leitora.
A versão corrente do firmware até a impressão deste manual é, conforme a figura abaixo: **CAS-EMV101U-0097**.

Figura B.2 - Identificação da versão de firmware da leitora Smartnonus

B.3) Identificação do Software.

O software de demonstração de API Nonus possui a identificação da versão na sua janela principal.
Na figura B.3.1, é possível observar esta identificação como "Demonstração API Smartnonus v.1.0.
Para identificar a versão do software ou driver instalado, é preciso acessar o Gerenciador de Dispositivos, para isto:

a) Aponte e clique no menu Iniciar.
b) Em seguida aponte e clique em Painel de Controle
c) Aponte e clique na categoria Sistema e Segurança
d) Em seguida aponte e clique em Sistema
e) Na aba esquerda, clique agora em Gerenciador de Dispositivos.

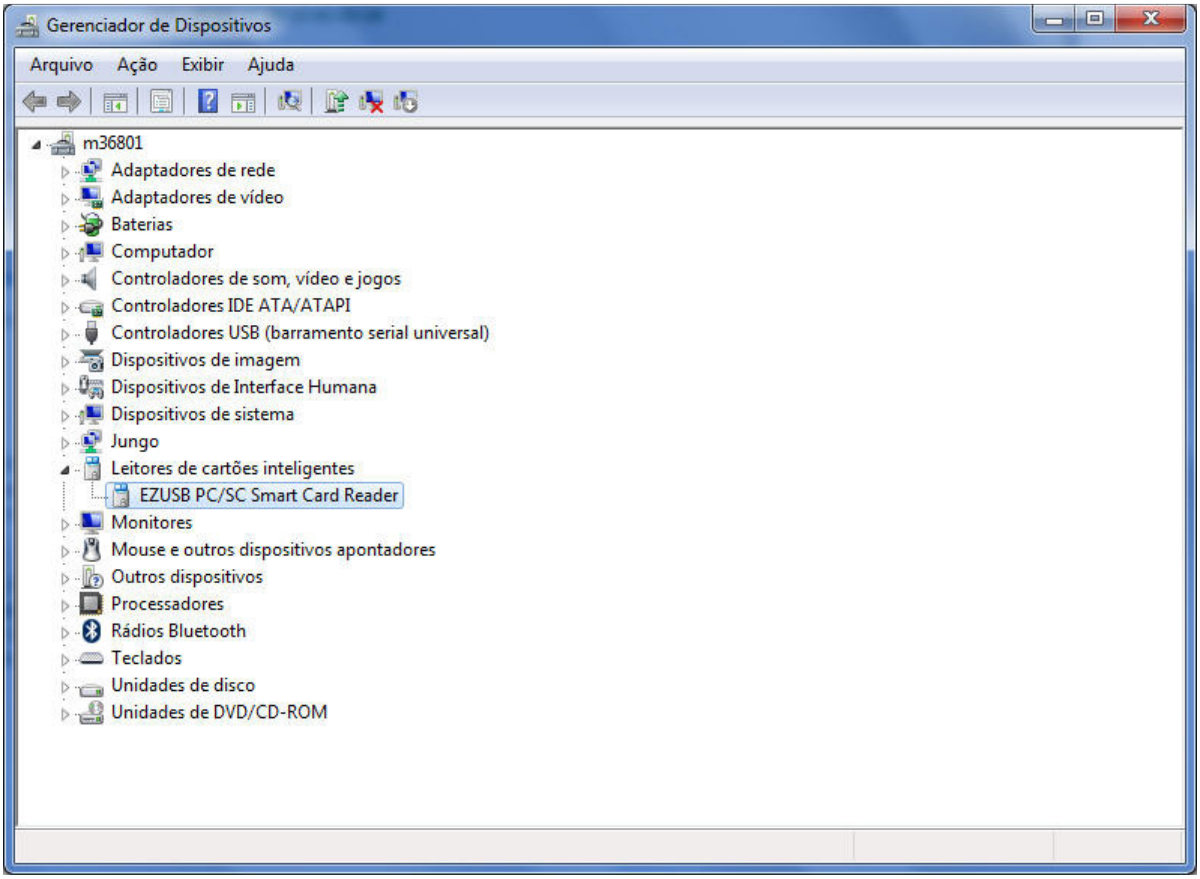


Figura B.3.1 - Gerenciador de Dispositivos

f) Localize e clique no grupo Leitores de cartões inteligentes, clique em "EZUSB PC/SC Smart Card Reader" com o botão direito do Mouse e aponte e clique em Propriedades.
g) Clique na aba driver.

Nesta aba esta identificada a versão corrente do driver, conforme a figura abaixo:
A versão corrente do software driver até a impressão deste manual, para o Windows 7, é conforme abaixo: **3.1.7.0**.

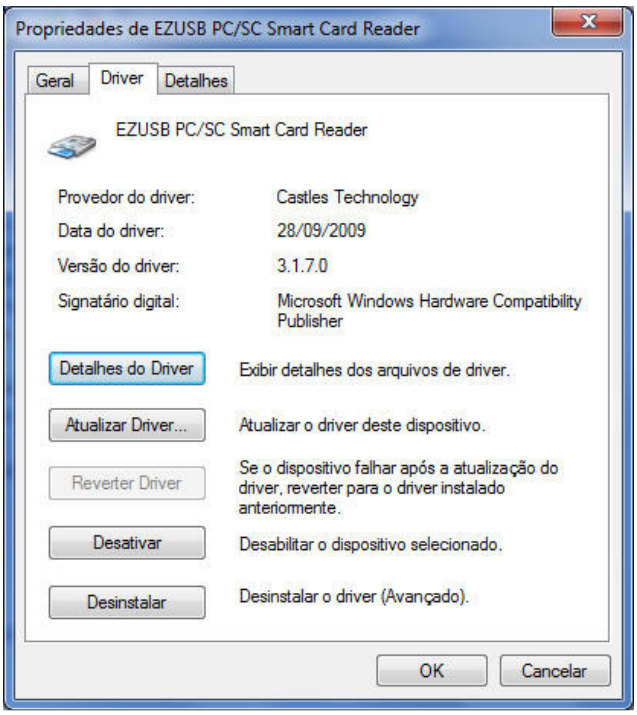


figura B.3.2 - Aba Driver de Propriedades da leitora.

C) Demonstração API Smartnonus

A Nonus fornece uma demonstração de funcionamento da API da leitora Smartnonus.
Este programa tem duas finalidades; permite testar a instalação da leitora Smartnonus e fornece para desenvolvedores de aplicações para cartões inteligentes um modelo de desenvolvimento inicial.

A API Smartnonus esta disponível em 2 formas:

C.1) Aplicação CardReader

A aplicação na sua forma executável serve para aqueles que desejam testar a instalação da leitora Smartnonus em seu sistema, bem como obter informações sobre a versão de firmware e dados PC/SC da leitora. Pode ser obtida por download gratuito em www.nonus.com.br/smartnonus.
Descompacte e execute o arquivo CardReader.zip.

C.1.1) Aba Informações

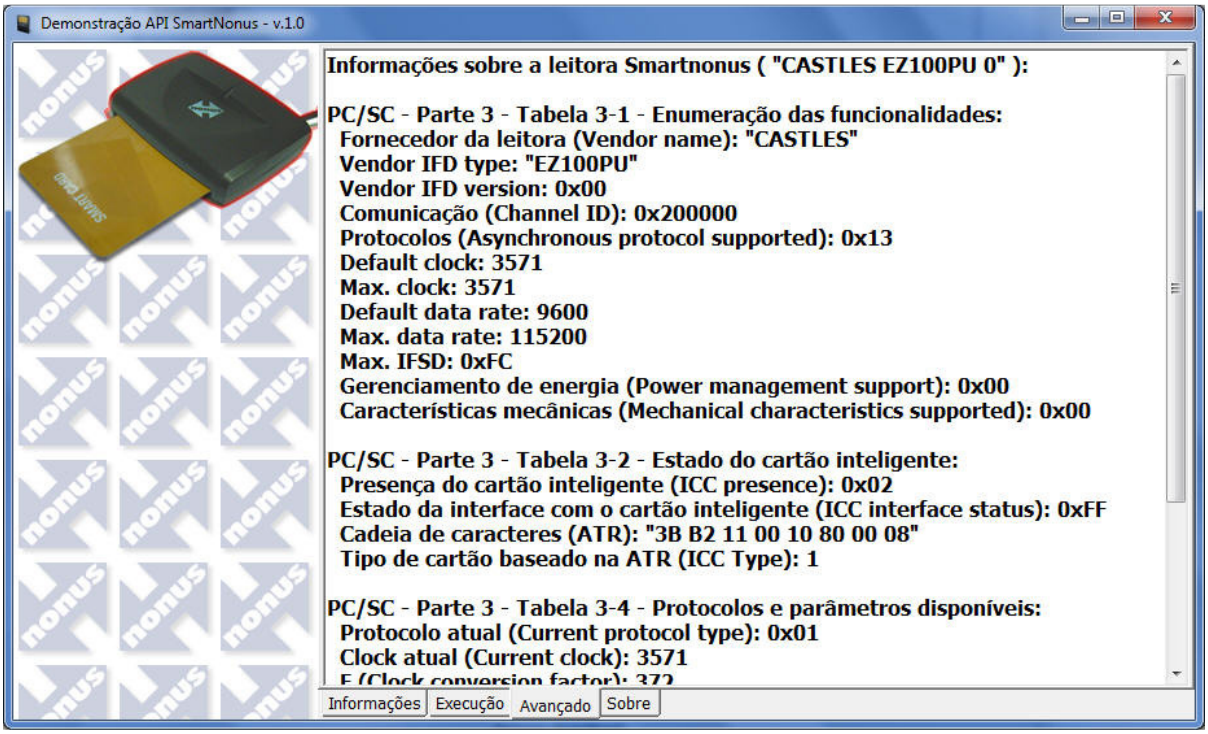
Fornece dados sobre: Versão do Firmware, estado da Leitora e ATR do Cartão.

C.1.2) Aba Execução

Fornece informações em tempo real sobre o estado da leitora

C.1.3) Aba Avançado

Fornece informações sobre as invocações via SP, de Fornecedor da Leitora, Comunicação, Protocolos, estado da interface com o cartão, ATR e outros.
Vide sessões 4.2, 4.4 e 4.5.1 deste manual.



C.2) Código Fonte

O código fonte está aberto e disponível para uso de desenvolvedores sem custos adicionais.
Pode ser adaptado, copiado e distribuído para uso com as leitoras Smartnonus.
Pode ser obtido por download gratuito em www.nonus.com.br/smartnonus.

Requisitos Mínimos de Ambiente Recomendados:
Ambiente de Desenvolvimento: Delphi 2010, XE (www.embarcadero.com)
Microsoft Windows 7
Processador: 1GHz (ou superior)
Memória: Mínimo de 2GB
Espaço em Disco: 250MB
Versão do Driver: 3.1.7.0 - API Smartnonus (winscard.dll), disponível para download em www.nonus.com.br/smartnonus
Manual da API Smartnonus, vide sessão 6 deste manual - Referência API PC/SC - Smartnonus

O código fonte pode ser adaptado para outras linguagens de programação mediante consulta ao suporte técnico Nonus.