OKI

SCOPE

Manual do Desenvolvedor

Informações Gerais

Leia antes de utilizar o software:

É imprescindível a leitura deste material para o correto procedimento de instalação, configuração e administração deste produto.

Revisão	Data	Responsável	Histórico
1.0	13/04/2010	Equipe Desenv. SCOPE	Versão Inicial.
1.1	30/04/2010	Equipe Desenv. SCOPE	Revisão do texto e inclusão de novos retornos e tipos de dados.
1.2	01/07/2010	FacTI – Vinicius Peracini	Inclusão de tabela com código das redes.
1.3	31/08/2010	Paulo dos Santos	Inclusão da máscara Logo_PAN (usada na ScopeObtemCampoExt).
1.4	04/10/2010	FacTI – Tiago Valério Frederico C. Goulart Jailton da S. C. Santana	Inclusão dos códigos de retorno (coleta de dados), da descrição do novo tipo de dado, inclusão da nova função na sessão “Funções diversas” - obtendo os campos, inclusão do código de bandeira, código do grupo de serviços e código do serviço, referentes à Plataforma Promocional da rede Cielo. Novo tipo no ScopeForneceCampo. Alterações referentes às transações Banrisul EMV. Acréscimo de tópico para detalhes específicos das redes.
1.5	29/12/2010	Paulo dos Santos	Atualização da tabela de bandeiras.
1.6	05/01/2011	Marcos Teruhiro	Atualização da tabela de bandeiras.
1.7	16/02/2011	Jailton da S. C. Santana	• Inclusão da máscara para a função ScopeObtemCampoExt2; • Inclusão dos novos serviços de pagamento, DARF e GPS; • Inclusão dos novos estados para interrupção e coleta de dados. • Exclusão da bandeira GreenCard • Atualização da seção *Configurações gerais* e *Tratando queda de energia*. • Inclusão das bandeiras Goodcard Senha, NEUS SENHA, Nutricash e Diamante. • Revisão do texto referente à reimpressão de comprovantes. • Inclusão da função ScopeServicoTecnico. • Inclusão da chave WKPAN na configuração do arquivo scope.ini.
1.8	17/03/2011	Paulo dos Santos	Atualização da tabela de bandeiras.
1.9	18/03/2011	Paulo dos Santos	Inclusão das Máscaras DescResgateMonetario e Tipo_Cartao para uso na função ScopeObtemCampoExt2.
1.10	19/04/2011	FacTI – Daniel Alves Paraiso	Atualização da tabela de redes e bandeiras.
1.12	26/04/2011	FacTI – Rafael D. Tardivo	Inclusão de Rede / Bandeira DMCARD em suas respectivas tabelas.

1.13	29/04/2011	FacTI – Felipe M. Schaden	Inclusão da Rede Siscred / Bandeiras Siscred e Planvale.
1.14	04/05/2011	FacTI – Vinicius Peracini	Atualização da tabela de códigos de erro.
1.15	11/05/2011	FacTI – Felipe M. Schaden	Inclusão da Bandeira DMCARD SENHA.
1.16	07/06/2011	FacTI – Vinicius Peracini	Atualização da tabela de redes e bandeiras.
1.17	02/08/2011	FacTI – Juliana Bersi	Atualização da tabela de códigos de erro com o erro 0xFF6B. Inclusão do protótipo das funções: ScopeMenuSelecionaItem e ScopeMenuRecuperaItens.
1.18	09/08/2011	Frederico C. Goulart	Atualização da tabela de redes e bandeiras.
1.19	23/08/2011	FacTI – Rafael D. Tardivo	• Coleta/Confirmação do valor do saque pelo PIN-pad. • Inclusão de índices/máscaras nas funções ScopeForneceCampo, ScopeSuspend, ScopeConfigura e ScopeObtemCampoExt2 para uso em configurações de coleta/validação do saque. Inclusão da chave “ScopeValidaSaque”, configurável por chave no .INI ou REGISTRY.
1.20	12/09/2011	Marcelo Dionysio da Fonseca	• Consulta Crédito Infocards. • Nova coleta: TC_COLETA_TIPO_CONSULTA (0xFCB8) • Novo campo da Mascara 3 do ObtemCampo: Resposta_Consulta_Infocards (0x00000020) 0x0000002
1.21	15/09/2011	FacTI – Juliana Bersi	• Inclusão dos três novos estados de coleta (0xFCB6, 0xFCB7, 0xFCB9), para quando for solicitado o cartão digitado no Pin Pad.
1.22	04/11/2011	Edison Ogata	• Inclusão dos estados de coleta 0xFCBA e 0xFCBB para o Troco Visa
1.23	29/11/2011	Douglas Sebastiani Bucciolli	• Inclusão dos tópicos Plataformas Compatíveis, Requisitos de HW e Requisitos de SW • Atualização do glossário.
1.24	5/01/2012	Facti – Rafael D. Tardivo	• Inclusão dos códigos de coleta de Coleta DDD + Telefone no PIN Pad e Redigita DDD + Telefone no PIN Pad
1.25	20/01/2012	FacTI – Sérgio R. A. Assunção Jailton da S. C. Santana	• Nova interrupção: EC_COLETA_DADOS_ECF • Nova coleta: TC_COLETA_DADOS_ECF (0xFCB9) • Nova bandeira: Ticket Car • Nova função: ScopeAtualizaParametrosChip • Novo código para ScopeForneceCampo • Acrescentado tópico da rede Ticket Edenred • Acrescentada a rede e a bandeira EPay
1.26	05/03/2012	Facti – Felipe M. Schaden Facti – Tiago F. Valério	• Alteração da bandeira PL-COSAN para PL GETNET VISA. • Inclusão da Bandeira PL GETNET MASTER.
1.27	23/03/2012	Edison Ogata	• Inclusão da seção PINPAD no arquivo scope.ini
1.28	16/04/2012	Facti – Tiago F. Valério	• Inclusão da rede TENDENCIA.

1.29	26/04/2012	Facti – Rafael D. Tardivo	• Inclusão do detalhamento sobre as configurações de registro do SCOPE no WoW64 • Inclusão da sessão “Fontes de Dados do Scope no WoW64”
1.30	11/05/2012	Facti – Rafael D. Tardivo	• Inclusão da nova estrutura de Dados de Recarga de Celular “stREC_CEL_VALORES_MODELO_4”
1.31	22/05/2012	Facti – Felipe M. Schaden	• Inclusão da Bandeira POLICARD SENHA
1.32	20/06/2012	Edison Ogata	• Inclusão da função ScopeVersao
1.33	21/06/2012	Facti – Felipe M. Schaden	• Inclusão da rede RVTECNOLOGIA.
1.34	03/07/2012	Edison Ogata	• Inclusão da função ScopeObtemDadosTotalTEFEx() e atualização dos tipos de *layouts* que podem ser utilizados por ela.
1.35	08/08/2012	Marcelo D. Fonseca	• Nova função: Simulação Crediário; • Novos serviços: Crediário e Simulação Crediário.
1.36	10/08/2012	Facti – Tiago F. Valerio	• Inclusão da função adicional e nova lista de medicamentos na sessão sobre PBM. • Incluido novo código na lista de códigos de erro do Scope.
1.37	30/08/2012	Facti – Tiago F. Valerio	• Inclusão de novo estado de coleta de dados TC_DECIDE_PGTO_CARNE e estado para interrupção EC_DECIDE_PGTO_CARNE.
1.38	13/09/2012	Edison Ogata	• Inclusão do código 32 para a função ScopeConfigura() permitindo a gravação dos dados da coleta em arquivo. • Inclusão do valor 8 para a função ScopeObtemHandle() que recupera dados da coleta gravados em arquivo. Valores serão retornados somente se o código 32 estiver habilitado ScopeConfigura().
1.39	19/09/2012	Facti – Felipe M. Schaden	• Inclusão da rede CREDITEM e bandeira CREDITEM SENHA.
1.40	21/09/2012	Edison Ogata	• Atualização das tabelas de redes e bandeiras.
1.41	15/10/2012	Facti – Rafael D. Tardivo	• Inclusão do ID para obtenção dos valores retornados na resposta de consulta de saldo da EPAY na máscara 3 da função ScopeObtemCampo
1.42	17/10/2012	Facti – Tiago F. Valerio	• Inclusão da rede GIVEX, bandeira GIVEX, atualização das sessões Estados para interrupção e Coleta de dados.
1.43	03/01/2013	Edison Ogata	• Atualização da "Sessão [PINPAD]" com informação da limitação de dígitos de coleta do pinpad. • Atualização da "Recarga de celular" com informações de parametrização de coleta.
1.44	04/01/2013	Facti – Felipe M. Schaden	• Inclusão da bandeira Sodexo Refeicao
1.45	05/03/2013	Edison Ogata	• Inclusão da bandeira ABRAPETITE
1.46	08/03/2013	Edison Ogata	• Acréscimo da seção ArqControlPath no scope.ini
1.47	13/03/2013	Facti – Henrique de O. Buzachero	• Inclusão da rede e bandeira TOPCARD
1.48	18/04/2013	Edison Ogata	• Inclusão do parâmetro MascararDados
1.49	07/05/2013	Facti – Henrique de O. Buzachero	• Inclusão da rede VR e bandeiras da VR
1.50	25/06/2013	Facti – Camila M. Dutra	• Inclusão da rede e bandeira INCOMM, reutilizando o código de rede e bandeira BANKBOSTON. • Inclusão de Apêndice sobre o formato do código de barras da INCOMM tratado pela automação comercial.
1.51	01/07/2013	Facti - Felipe M. Schaden	• Inclusão da descrição de novas funcionalidades de Atualização de Preços e Fornecimento de Lista de Mercadorias para a Ticket Car

1.52	01/07/2013	Regina M. N. Sato	• Inclusão da máscara da função ScopeObtemCampoExt2 para obtenção do Maximo de Mercadorias permitidas para transação TicketCar.
1.53	06/08/2013	Facti – Juliana Bersi	• Inclusão da rede e bandeira ELAVON
1.54	11/10/2013	Facti – Camila M. Dutra Facti - Felipe M. Schaden	• Inclusão das bandeiras Bancred Senha e Vegas Card Senha
1.55	14/10/2013	Facti – Felipe M. Schaden	• Inclusão da rede VR e das bandeiras VR e Multibeneficios • Descrição do ForneceCampo SCOPE_DADOS_APLIC_CBD
1.56	13/01/2014	Jovelina M. de Jesus	• Inclusão de rede e bandeira PharmaSystem; funções: Elegibilidade, Pré-Autorização, Cancelamento de Pré-Autorização; layouts de lista de medicamentos; função para recuperação de lista de projetos.
1.57	11/02/2014	Jovelina M. de Jesus	• Inclusão das rotinas de PIN-Pad para leitura e limpeza de comanda. Inserção de dois códigos de retorno de erro para PIN-Pad compartilhado.
1.58	21/03/14	Jailton da S. C. Santana	• Descrição da funcionalidade de transação POS • Inclusão da configuração do scope.ini para a transação POS
1.59	25/03/2014	Edison Ogata	• Inclusão da bandeira FORTBRASIL
1.60	02/06/2014	Facti – Felipe M. Schaden	• Inserido item Consulta a saldo de cartão de débito • Corrigido código para TC_COLETA_DADOS_ECF • Incluida informação de Consulta das Mercadorias configuradas pela Inicialização de Tabelas da Ticket Edenred. • Complemento e correção de informações específicas para a rede Ticket.
1.61	04/07/2014	Facti-Camila M. Dutra	• Inclusão de códigos de Convênio, Rede e Bandeira da PBM PADRÃO • Atualização da sessão “Compra de medicamento” a respeito do funcionamento do layout 5 • Atualização da sessão "Lista de medicamentos Ex"
1.62	07/07/2014	Facti – Felipe M. Schaden	• Inclusão da coluna com os Códigos SAT na tabela Código das redes. • Inclusão da máscara da função ScopeObtemCampoExt2, na tabela Dados Disponíveis, para obtenção do código SAT da transação.
1.63	07/07/2014	Facti – Felipe M. Schaden	• Melhoria do detalhamento do campo SCOPE_DADO_SEPARADOR_LINHA da função ScopeForneceCampo() para parametrização do finalizador de linha do cupom • Informações adicionais para formatação do finalizador de linha da função ScopeGetCupomEx.
1.64	08/07/2014	Alcimar Joaquim	• Inclusão dos possíveis erros retornados pelas as funções do Scope. • Inclusão dos códigos de coleta o Scope para as funções: ScopeCompraCartaoCredito, ScopeCompraCartaoDebito, ScopeRecargaCelular e ScopeAtualizaPrecosMercadorias.
1.65	14/07/2014	Alcimar Joaquim	• Inclusão do tópico “Cielo” na seção “Redes com tratamentos específicos” • No tópico “Cielo” está descrito como implementar as funcionalidades do Cielo Premia.
1.66	30/07/2014	Alcimar Joaquim	• Inclusão da função ScopeObtemOperadorasRecCelOffTEF • Inclusão da função ScopeRecuperaValoresRecCelOffTEF
1.67	26/08/2014	Marcelo Dionysio da Fonseca	• Inclusão da rede Bin

1.68	27/08/2014	Facti – Felipe M. Schaden	• Inclusão de seção especifica para detalhamento da rede SAVS • Descrição da função ScopeRecuperaBufTabelaGen para SAVS • Inclusão de Códigos de retorno para SAVS • Inclusão de Estados para interrupção para SAVS • Inclusão de Código de Rede e Bandeira SAVS • Inclusão de Código de Serviço para SAVS • Inclusão de Constante Versao_Carga_Tabelas_Host da máscara 3 da função ScopeObtemCampoExt2
1.69	28/08/2014	Edison M. Ogata	• Inclusão da rede CONDUCTOR-PL • Inclusão da bandeira BAHAMAS CRÉDITO • Inclusão da bandeira BAHAMAS ALIMENTAÇÃO
1.70	23/09/2014	Facti – Felipe M. Schaden	• Correção de nome da função ScopeRecuperaBufTabelaGen e de parâmetro da função ScopeRecuperaBufTabela
1.71	19/12/2014	Regina Megumi	• Inclusão de informações sobre Cielo Auto
1.72	22/12/2014	Marcelo Dionysio da Fonseca	• Atualização para Transações Sem Contato (Contactless) • Inclusão da tabela “Códigos de Retorno de Pré-TEF” no Apêndice A.
1.73	27/01/2015	Edison Ogata	• Acréscimo da bandeira CABAL VOUCHER na lista de bandeiras.
1.74	19/02/2015	Edison Ogata	• Reitrada da bandeira CABAL VOUCHER da lista de bandeiras, pois já existe uma bandeira semelhante: CABAL VALE
1.75	27/02/2015	Facti – Felipe M. Schaden	• Inclusão da máscara da função ScopeObtemCampoExt2, na tabela Dados Disponíveis, para obtenção do código SAT e CNPJ da rede credenciadora da transação.
1.76	12/03/2015	Jailton Santana	• Inclusão do parâmetro GenTiemout • Formatação

Índice

Prefácio 13
A quem se destina 13
Requisitos 13
Plataformas Compatíveis 13
Requisitos de HW 13
Requisitos de SW 13
Descrição do Produto 13
Objetivo 13
Definição 13
Benefícios 13

Conceitos 15
Transações 15
Fluxo de estados de coleta 16
Interfaces de interação com o SCOPE Client 16
Interface coleta 16
Interface HLAPI 17
Padrões adotados neste documento 17
Legenda de Abreviações 19

Instalação 20
Instalação do SCOPE Client para MS-Windows® 20
Instalação do SCOPE Client para Linux 20

Configuração 22
Configuração do arquivo scope.ini 22
Sessão [<empresa><filial>] 22
Sessão [SCOPEAPI] 24
Sessão [PPCOMP] 24
Configuração de porta serial para Linux 24
Sessão [PINPAD] 25
Seção [SCOPEAPIPOS] 25
Seção [SCOPEAPIPOS-RXXX] 26
Configuração do registro do MS-Windows® 27
Fontes de Dados do SCOPE no WoW64 28

Funções básicas da API do SCOPE Client 29
Comunicação com o ScopeSRV 29
Conexão 29
Desconexão 30
Sessão de transação 31
Sessão de TEF 31
Abrindo uma sessão 32
Encerrando a sessão 33
Tratando queda de energia 34
Deixando transação pendente na queda de energia 35
Status de transação 36
Consultando o status 36

Funções de configuração de ambiente 38
Configurações gerais 38
Configurando em tempo de execução 39
Configuração de PIN-Pad 40
Validando a interface de PIN-Pad 40

Funções específicas das interfaces 41
Interface coleta 41
Definindo a interface coleta 41

Obtendo os parâmetros da transação 43
Passando o dado da coleta ao SCOPE Client 45
Interface HLAPI 48
Suspendendo a transação 48
Retomando a transação 50
Abortando a transação 51

TEF 53
Cartão de crédito 53
Compra com cartão de crédito 53
Consulta a financiamento de cartão de crédito 55
Consulta a saldo de cartão de crédito 56
Pré-autorização de crédito 57
Cartão de débito 59
Compra com cartão de débito 59
Compra CDC (Crédito Direto ao Consumidor) 61
Consulta CDC 61
Simulação Crediário 62
Consulta a saldo de cartão de débito 64

Cartão Dinheiro 65
Operações 65
Compra/Carga do cartão dinheiro 65
Compra usando o cartão dinheiro (débito) 66
Consulta ao saldo de cartão dinheiro 67
Estornos 67

Funções de Consulta 68
Cheque 68
Consulta de cheques 68
AVS 69
Consulta AVS 69

Recarga de celular 71
Configurando a recarga de celular 71
Processando a recarga de celular 71
Iniciando a transação de recarga 72
Obtendo operadoras disponíveis 73
Obtendo valores de recarga 76
Funções de consulta para recarga de celular 81
Obtendo operadoras disponíveis fora do fluxo de recarga 81
Obtendo valores de recarga disponíveis fora do fluxo de recarga 82

Estorno de transações 84
Estornando a transação 84

Comprovantes 86
Comprovantes de transações 86
Obtendo os cupons de TEF 86
Reimpressão de comprovante 88
Solicitando o comprovante off-line 88
Solicitando o comprovante on-line 89
Imprimindo o comprovante correto 91

PBM - Medicamentos 94
Consultando medicamento 94
Compra de medicamento 95
Lista de medicamentos 97
Lista de medicamentos com CRM 98
Lista de medicamentos Extendida 100
Lista de Projetos 101
Elegibilidade do Cartão PBM 102

Pre-Autorizacao de Medicamentos PBM 104
Cancelamento de Pre-Autorizacao de Medicamentos PBM 105

Funções de PIN-Pad 107
Comunicação com o PIN-Pad 107
Iniciando a comunicação 107
Encerrando a comunicação 108
Mensagens no visor 108
Mensagens de tamanho padrão 108
Mensagens maiores 109
Informações obtidas do PIN-Pad 110
Dados das redes autorizadoras 110
Dados do dispositivo 110
Obtendo as informações 111
Obtendo teclas do PIN-Pad 112
Iniciando a leitura da tecla 112
Obtendo a tecla 112
Obtendo a senha aberta 113
Iniciando o modo de leitura da senha aberta 113
Lendo a senha aberta 114
Obtendo a senha criptografada 115
Iniciando o modo de leitura da senha criptografada 115
Lendo a senha criptografada 116
Lendo um cartão 117
Iniciando a leitura do cartão 117
Recuperando o cartão 117
Abortando o processamento (*ScopePPAbort*) 119
Mensagens de erro (*ScopePPMsgErro*) 120
Obtendo a comanda de um cartão 121
Iniciando o modo de leitura da comanda 121
Lendo a comanda 122
Limpando a comanda de um cartão 123
Iniciando o modo de limpeza da comanda 123
Limpando a comanda 123

Totalização de TEF 125
Relatório de TEF 125
Iniciando os totais 125
Cupom dos totais 126
Dados dos totais 128

Transação de POS para conciliação 132
Descrição da funcionalidade 132
Transação POS 132
Redes e bandeiras 135
Códigos de serviços 139
Cupom CIELO X ScopeObtemCampoExt2 X arquivo 140
Cupom REDECARD X ScopeObtemCampoExt2 X arquivo 141
Outros cupons X ScopeObtemCampoExt2 X arquivo 143
Formatação para o arquivo de conciliação 144

Funções diversas 145
Dados da transação 145
Obtendo *handle* 145
Obtendo os campos 146
Fornecendo informações extras para a transação 150
Descrição dos tipos de dados 150
Atributos da aplicação 152
Atrelando dados à transação 153
ScopeGetLastMsg 155
ScopeGetCheque 156

ScopeAtualizaValor 156
ScopeGarantiaDescontoCheque 157
ScopeTransacaoFinanceira 157
ScopeInvestimento 158
ScopeObtemCartaoInvestimento 159
ScopeResumoOperacoes 160
ScopePagamento 161
ScopeServicoTecnico 162
ScopeAtualizaParametrosChip 163
ScopeVersao 164
ScopeAtualizaPrecosMercadorias 165

Redes com tratamentos específicos 167
Banrisul EMV 167
PINPads com suporte a rede 167
Associação dos produtos e as funções do SCOPE 167
Transação convênio combustível 167
Ticket Edenred 168
PINPads com suporte a rede 169
Associação dos produtos e as funções do SCOPE 169
Dados coletados 170
Consulta saldo 170
Dados do ECF 170
Dados da Lista de Mercadorias Consumidas 171
Descrição do formato do registro "SC101": 172
Dados da Lista de Atualização de Preços de Mercadorias 172
Comprovantes 173
Código das Mercadorias 173
Cielo Premia 174
Como é a gestão da campanha? 174
Quais tipos de prêmios o comprador pode receber? 174
Que tipos de transações podem receber os prêmios? 175
É possível restringir para tratar apenas prêmios não monetários, ou seja, não permitir descontos? 175
O que é preciso configurar no SCOPE CNF? 175
A Cielo precisa habilitar o Cielo Premia? 175
É possível integrar e testar através do SCOPE Autorizador? 175
É possível fazer um teste usuando o HotKey? 176
A Automação Comercial deve se adequar? 176
O que a Automação Comercial precisa fazer para estar aderente? 176
Como saber se houve um prêmio não monetário (brinde) recebido durante uma compra? 177
Como fazer um Resgate de Prêmio? 178
Existe uma forma de tratar os itens do Menu Dinâmico pela aplicação de AC? 180
Existe um código exemplo? 180
Qual é a versão mínima do SCOPE que contempla Cielo Premia? 180
Cielo Auto 180
Lista de produtos e/ou serviços principais Cielo Auto 181
Lista de produtos e/ou serviços secundarios Cielo Auto 181
Informações Adicionais Cielo Auto 184
SAVS 184
Associação dos produtos e as funções do SCOPE 184
Autorização de Vouchers e Serviços 185
Estorno de Autorização de Vouchers e Serviços 187
Lista de Fornecedores 187
Lista de Produtos 188
Cielo – Transações Sem Contato (Contactless) 190
Requisitos 190
Funcionamento 190

Apêndice A – Tabelas 192
Códigos de retorno 192
Coleta de dados 192

Autorizadoras 196
Códigos de erros do SCOPE 197
PIN-Pad compartilhado 200
Códigos de Retorno de Pré-TEF 202
Formatos dos dados 204
Códigos das Teclas 204
Códigos de Fluxo 205
Estados para interrupção 205
Código das redes 211
Código de especificação das redes 214
Código das bandeiras 214
Dados disponíveis das transações 219
Grupo de Serviços 221
Códigos dos Serviços 221
Convênios 223

Apêndice B – Especificação Visanet 4.1 225
Adequação 225
Certificação 226

Apêndice C – PIN-Pad Compartilhado 227

Apêndice D – Conjunto de bibliotecas do SCOPE Client 229
MS-WINDOWS® 229
Linguagem Java 229
Linux 230
Linguagem Java 230

Apêndice E – Identificando a versão do SCOPE Client 231
Verificando no SCOPE Server 231
Verificando no ambiente do PDV 232
SCOPE Client para MS-Windows® 232
SCOPE Client para Linux 233

Apêndice F – Formato do Código de Barras InComm 235

Apêndice G - Glossário 236

Prefácio

A quem se destina

O manual do integrador destina-se aqueles que desenvolvem aplicação em que é necessário efetuar transações com autorizadoras, banco, etc. e para isso utilizarão o SCOPE como concentrador de TEF.

Requisitos

Plataformas Compatíveis

Todos os módulos da solução SCOPE, são compatíveis com as plataformas Windows 7 e Windows Server 2008.

O módulo SCOPE Client, além das plataformas acima, também é compatível com a plataforma Linux (Librix, RedHat, SUSE e outros).

Requisitos de HW

- PC com processador de 1 GHz ou mais de velocidade
- 512 megabytes (MB) de RAM ou mais são recomendados
- 1 gigabyte (GB) de espaço disponível em disco rígido ou mais são recomendados

Requisitos de SW

Banco de Dados (SGBD). Os seguintes SGBDs são suportados pela solução SCOPE:

- SQL Server 2005 ou 2008
- Oracle 10g ou 11g

Descrição do Produto

Objetivo

O sistema **SCOPE – Solução Completa para Pagamento Eletrônico** – permite ao software de **PDV** efetuar o pagamento através de **TEF**, por exemplo, cartões de crédito e débito, através da digitação do emboço ou leitura da tarja magnética ou chip.

Definição

O **SCOPE** foi concebido para gerenciar todas as etapas de uma TEF abstraindo a complexidade inerente à transação do aplicativo PDV.

Benefícios

- O sistema permite uma arquitetura flexível, multi-empresa, multi-filial, centralizada ou distribuída, adequando-se à necessidade do cliente.
- Facilita a gestão do negócio, pois oferece ferramentas de consulta e relatórios operacionais e gerenciais.
- Permite a gestão da sessão contábil, pelo estabelecimento ou pelo centro de processamento, de forma a indicar eventuais divergências nos créditos efetuados pelos bancos.
- Permite mecanismos de alta-disponibilidade através de solução cluster e rotas de contingência.

Conceitos

Transações

O SCOPE provê à loja diversos tipos de transações (exemplo: compra com cartão de crédito, compra com cartão de débito, recarga de celular, pagamento de contas e de fatura de cartões, etc.). Numa transação, o SCOPE Client comunica-se com o SCOPE Server baseado em mensagens, as quais seguem a norma ISO 8583 que especifica o protocolo de mensagens para transações financeiras com cartão. As mensagens básicas e as suas sequências que geralmente compoem uma transação completa e bem sucedida são demonstradas na figura 1 e descritas abaixo. Observe que a informação apresentada abaixo não é exposta à interface de programação, compreendendo um fluxo interno entre SCOPE Client e SCOPE Server.

- 9604 – mensagem de solicitação da pré-TEF: esse tipo de mensagem é geralmente enviado duas vezes com finalidade diferentes (primeira e segunda pré-TEF). A primeira pré-TEF contém o BIN de um cartão e um grupo de serviço além de outros dados exigidos numa mensagem ISO. A segunda abrange o produto selecionado.
- 9614 – mensagem de resposta da pré-TEF. Contém os dados solicitados pelo SCOPE Client. Como a mensagem 9604, esta diferirá de acordo com o contexto da pré-TEF. A resposta da primeira é a rede autorizadora, a bandeira, os serviços disponíveis e os atributos (ex.: limite de parcelas, data limite de agendamento, etc.) dos produtos habilitados para o cartão capturado. A da segunda define para o SCOPE Client como as mensagens das transações deverão ser montadas.
- 0200 – mensagem de solicitação da transação financeira. Neste momento, a mensagem vai até a rede autorizadora para que seja autorizada.
- 0210 – mensagem de resposta da transação financeira. A autorizadora responde a solicitação com esta mensagem que abrange a aprovação ou a rejeição da solicitação.
- 0202 – mensagem de confirmação da transação financeira. Esta mensagem é a que confirma que a transação foi realmente efetivada. Ela indica, por exemplo, que o comprovante foi fisicamente impresso e que o processo como um todo foi concluído.

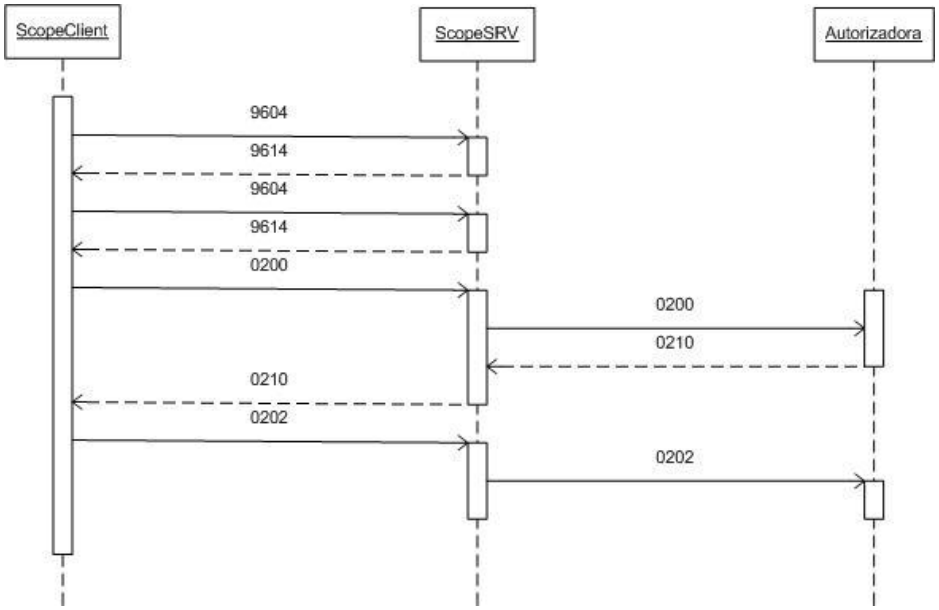


Figura 1: sequência de mensagens numa transação completa e bem sucedida.

Fluxo de estados de coleta

O funcionamento do SCOPE Client baseia-se em fluxos de estados de coleta para as diversas operações disponíveis. A cada iteração entre o aplicativo com o SCOPE Client, este informa em qual estado de coleta se encontra, qual informação deve ser coleta, qual mensagem deve ser exibida e quais opções de fluxo estão disponíveis (retornar, próximo e cancelar). Desta forma o fluxo avança até que seja possível efetuar a transação desejada.

Exemplificando: Ao iniciar uma transação de crédito, o SCOPE estará no estado de “coleta número do cartão”. Uma vez obtido o cartão, o estado avança e dependendo de uma série de variáveis, poderá migrar para o estado “coleta os 4 últimos dígitos do cartão”. Conforme as configurações possíveis para o tratamento de serviço de crédito, o próximo estado poderá ser “coletar se à vista ou não”, o fluxo avança até a impressão do comprovante.

Interfaces de interação com o SCOPE Client

Antes de começar o desenvolvimento da integração com o SCOPE, é de fundamental importância a decisão de como a aplicação irá interagir com o SCOPE Client, escolhendo uma das interfaces disponíveis: interface coleta ou interface HLAPI. A escolha do tipo de interface depende de fatores tais como: Tempo de implementação, flexibilidade e plataforma adotada.

Interface coleta

Disponível para todos os sistemas operacionais com os quais o SCOPE Client é executado, a interface coleta do SCOPE Client é o que apresenta uma maior interação entre a aplicação de PDV e o SCOPE. Através desta interface, a aplicação se torna responsável pela coleta dos dados digitados pelo operador ou cliente conforme a solicitação a cada iteração do SCOPE. Como a

aplicação coletará os dados, ela se responsabilizará pela exibição da mensagem na tela e a entrada de dados para o operador, sendo que para alguns casos, deverá tratar também a limitação do tamanho do campo aceitável (ex.: para a coleta do número de segurança do cartão, a aplicação permitirá a entrada de um valor com no mínimo 3 e no máximo 5 dígitos).

Características da interface coleta

- Interceptar, para uso da própria aplicação, os dados coletados via digitação para o SCOPE;
- Não interfere na interface gráfica do usuário;
- Disponível para qualquer linguagem e sistema operacional;
- A aplicação deve ter um maior nível de "especialização", portanto, maior lógica na integração.

Interface HLAPI

Esta interface exige um nível menor de integração com a aplicação, pois toda a coleta de dado é realizada pelo SCOPE através de uma pequena janela, do próprio SCOPE, que é exibida na tela (figura 2). Está disponível apenas para aplicações MS-Windows® de 32-bits.

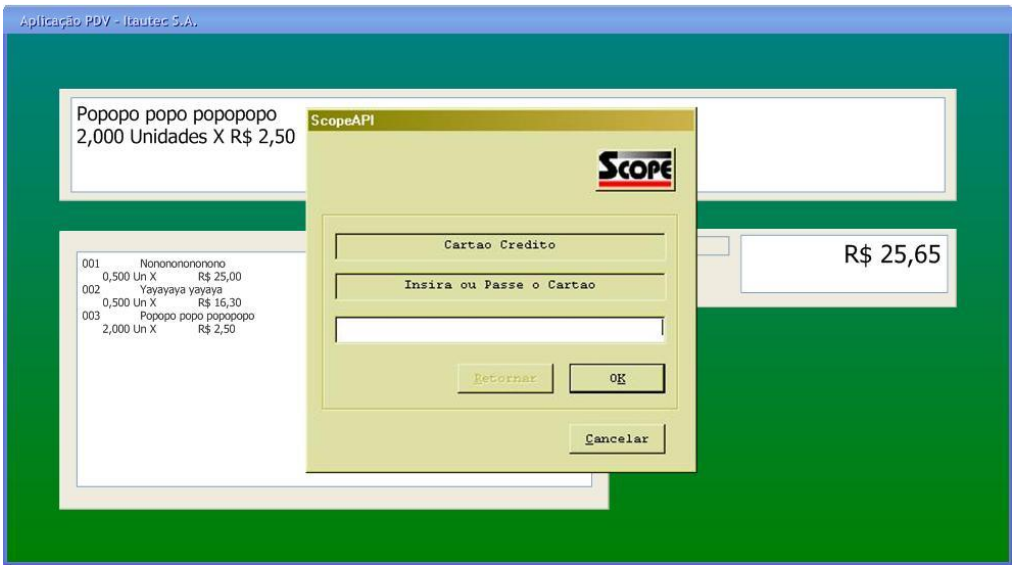


Figura 2: janela do SCOPE Client na interface HLAPI

Características da interface HLAPI

- A coleta é realizada pelo SCOPE, portanto a aplicação não sabe o que está sendo coletado;
- Disponível apenas para aplicações Win32;
- Pode tratar periférico (CMC7, impressora, etc.), desde que suportado pelo SCOPE;
- Possui GUI própria, que no momento da transação, fica aberta, sobrepondo a interface da aplicação.

Padrões adotados neste documento

Este documento descreve funções padronizadas no seguinte formato:

```
LONG EXPORT ScopeNomedaFuncao (argumentos...)
```

As sessões que descrevem os argumentos ou parâmetros das funções exibirão os mesmos numa tabela, cujas linhas representam cada parâmetro e as colunas indicarão os seguintes dados:

- 1ª coluna: parâmetro de entrada [in] ou saída [out];
- 2ª coluna: tipo de dado que representa o parâmetro;
- 3ª coluna: nome da variável representada no parâmetro
- 4ª coluna: significado do parâmetro.

Todos os exemplos de códigos relacionados neste documento estão seguindo a linguagem C com a seguinte formatação:

```
int main ()
{
   printf ("Hello, SCOPE\'s Programmers!");
}
```

Para compatibilidade das funções entre diversas linguagens e plataformas, algumas convenções de tipos devem ser adotadas, conforme a tabela a seguir:

Definição de Tipo:

Tipo	Tamanho	Descrição
BYTE	1 byte	Valor sem sinal de 0 a 255
WORD	2 bytes	Valor sem sinal de 0 a 65.535
SHORT	2 bytes	Valor com sinal de -32.768 a 32.767
LONG	4 bytes	Valor com sinal de -2.147.483.648 a 2.147.483.647

Analogamente, como recurso de portabilidade, algumas constantes devem ser criadas e definidas diferentemente em cada plataforma, com o objetivo de utilizar os mesmos protótipos de funções conforme a tabela a seguir:

Constantes:

Tipo	Descrição
EXPORT	Usada nos protótipos das funções que são *exportadas*
CALLBACK	Função cujo endereço de entrada é fornecido a outras funções, de modo que estas possam utilizar internamente da primeira

Legenda de Abreviações

Formato	Atributo
a	Caracteres alfabéticos.
n	Caracteres numéricos.
an	Caracteres alfabéticos e numéricos.
ans	Caracteres alfabéticos, numéricos e especiais.
MM	Mês.
DD	Dia.
AA	Ano.
hh	Hora.
mm	Minuto.
ss	Segundo.
LLvar	Tamanho de um campo variável. Ex. Se o conteúdo do campo = “AB1234CD”, teremos para representá-lo: “08 AB1234CD” .
LLLvar	Tamanho de um campo variável. Ex. Se o conteúdo do campo = “AB1234CD”, teremos para representá-lo: “008 AB1234CD” .
10	Tamanho fixo de 10 caracteres.
..10	Tamanho variável de até 10 caracteres.
b	Representação binária dos dados.

Instalação

Nesta sessão é comentada a localização das bibliotecas do SCOPE Client em cada sistema operacional. Quanto à composição do SCOPE Client, deve-se consultar o Apêndice E – Conjunto de bibliotecas do SCOPE Client.

Instalação do SCOPE Client para MS-Windows®

As bibliotecas do SCOPE Client para MS-Windows® devem estar em um diretório acessível pela aplicação. A decisão da localização destas bibliotecas é tomada pela equipe que desenvolve a aplicação de PDV, porque a particularidade da linguagem e/ou ambiente de desenvolvimento tem impacto sobre o acesso a elas.
Normalmente, o SCOPE Client é colocado no mesmo diretório da aplicação integradora. No entanto, há casos em que ele está disponível na pasta do sistema `%windir%\System32`.

CUIDADO: devido à perda de controle sobre qual biblioteca o MS-Windows® faz referência, não é aconselhável que as bibliotecas estejam no diretório do MS-Windows® ou em qualquer subdiretório.

No CD de instalação do SCOPE, há possibilidade de instalar o SCOPE Client na máquina. Esta instalação executará os seguintes passos:

- criará um diretório com o SCOPE Client;
- copiará o executável do HotKey e seu atalho na área de trabalho;
- alterará o registro do MS-Windows® com os itens de configuração.

O SCOPE se utiliza de alguns arquivos de controle (no formato SXXXXYYYYZZZAAA.sc_), gerados em tempo de execução. Assim, em sistemas operacionais com conceito de multiusuários, como Linux e MS-Windows, existe a necessidade de permissão para escrita e leitura no diretório da aplicação. No caso do MS-Windows, tais arquivos serão gerados em 'C:\'.

Instalação do SCOPE Client para Linux

Para a instalação do SCOPE Client em ambiente Linux, é disponibilizado um arquivo no formato RPM. Este arquivo é um pacote do sistema RPM (Red Hat Package Manager), utilizado em várias distribuições Linux. O nome do arquivo liberado segue o padrão:

`itautec-scope-<versão>-1.i386.rpm`

Onde: <versão> representa a versão e o release do SCOPE Client. Para a sua instalação execute o comando:

`$ rpm –i itautec-scope-<versão>-1.i386.rpm`

Este comando instalará as bibliotecas na pasta `/usr/lib`.

IMPORTANTE: para a execução deste comando é necessário ter direitos do usuário `root`.

CUIDADO: em ambiente Linux, o usuário root tem o direito de realizar comandos que podem danificar o sistema.

LEMBRETE: podem-se acrescentar os parâmetros "`-vh`" para exibir o progresso da instalação.

Configuração do arquivo scope.ini

O arquivo **scope.ini** é responsável por parte da configuração da aplicação. Ele deve estar disponível com o SCOPE Client, independente do sistema operacional que a aplicação executa.

Sessão [<empresa><filial>]

Esta sessão é a única obrigatória para que o SCOPE Client possa se conectar ao ScopeSRV. Dentro dos colchetes deve haver uma sequência de 8 dígitos, sendo que os 4 primeiros representam o código da empresa e os demais, o código da filial cadastrado no SCOPE. Estes códigos devem ser iguais aos utilizados como parâmetros da função `ScopeOpen()`. Os valores das chaves configuráveis nesta sessão se encontram na tabela abaixo.

Chave	Significado	Valor	
Name	Endereço ou nome da máquina em que está o ScopeSRV.	I.P. ou nome da máquina do ScopeSRV	
Port	Porta de conexão do ScopeSRV.	Valor numérico (o padrão é 2046).	
AtualizaDataHora	Parâmetro que decide se o SCOPE Client deverá alterar a data e a hora da máquina para sincronizar com o ScopeSRV no momento da conexão.	N	se não deve sincronizar (padrão).
		S	caso deseja-se que sincronize.
MsgOperReduzida	Controla o tamanho máximo de colunas das mensagens que o SCOPE Client enviará para a aplicação para que esta exiba no teclado do operador ou display com esta finalidade.	N	40 colunas (padrão)
		s	20 colunas
		m	16 colunas
ShowCupom	Configura a exibição do cupom na tela. Utilizado apenas na interface HLAPI.	n	Não exibe
		s	Exibe (padrão)
TimeOutLogon	Configura o tempo, em segundos, de espera para a conexão do SCOPE Client com o ScopeSRV.	Valor numérico entre 10 e 60 (padrão é 10).	
TimeOutAdm	Configura o tempo, em segundos, de espera da resposta da autorizadora em cada transação.	Valor numérico entre 15 e 180 (padrão é 30).	
CupomReduzido	Configura se deve exibir o cupom reduzido na tela. Utilizado apenas na interface HLAPI.	n	Não exibir
		s	Exibir
VersaoAutomacao	Versão da automação passada pela área de certificação da Visanet.	Sequência de caracteres no formato RRAAAACCCC, sendo que: RR → release de certificação do	

		TEF AAAA → Nome da automação CCCC → Código do memorando xxx (o padrão é “01HOTK0000”)	
WKPAN	Configura se deve habilitar a Comunicação Segura com o PIN Pad.	n	Não habilitar
		s	Habilitar
ScopeValidaSaque	Configura se o Scope deve consistir o valor do saque digitado com os valores configurados nos limites de saque do produto, localizados no ScopeCnf.	n	Não habilitar
		s	Habilitar
MascararDados	Indica como o valor do PAN deve ser enviado para a aplicação.	n	Nenhum cartão é mascarado.
		s	Todos os cartões serão mascarados com zeros.
		s(0)	Todos os cartões serão mascarados com zeros.
		s(*)	Todos os cartões serão mascarados com asteriscos.
		p	Todos os cartões que seguem o padrão PCI DSS serão mascarados com zeros.
		p(0)	Todos os cartões que seguem o padrão PCI DSS serão mascarados com zeros.
		p(*)	Todos os cartões que seguem o padrão PCI DSS serão mascarados com asteriscos.
GenTimeout	Define um tempo máximo, em segundo, para o SCOPE Client gerar as chaves de criptografia utilizado na conexão.	Valor numério de 1 à 99 que representa o tempo em segundos. Se não tiver configurado ou o valor estiver inválido, será assumido o valor padrão é de 99 segundos. Neste caso, será praticamente impossível alcançar esse tempo e a chave ideal será gerada sempre.	

As chaves mínimas exigidas são as duas primeiras (Name e Port), enquanto as outras são opcionais e dependerá da situação de cada aplicação.

Exemplo: para a configuração de um PDV cadastrado na empresa 0001 e filial 0007, que irá se conectar ao ScopeSRV no IP 10.50.9.70, o *scope.ini* deverá ser configurado da seguinte maneira:

```
[00010007]
Name=10.50.9.70
Port=2046
VersaoAutomacao= "01CAPS0001"    ; valor fictício
```

Sessão [SCOPEAPI]

Esta sessão não é obrigatória e define os parâmetros abaixo relacionados ao ambiente:

Chave	Significado	Valor	
SaveCupom	Diretório para salvar o cupom	Diretório (padrão é "./")	
TracePin	Habilita o log do PIN-pad compartilhado. Em MS-Windows, será gerado no 'C:\', nos demais sistemas, no diretório corrente da aplicação, o arquivo 'TracePin.sc_'. O sistema de log é circular, assim quando o tamanho desse arquivo for 1 MB, ele será renomeado para 'TracePin.bak' e um novo 'TracePin.sc_' será criado.	n	Não habilita (padrão)
		s	Habilita
ArqControlPath	Diretório para salvar os arquivos de controle do Scope Client. Somente válido para o sistema operacional Windows	Diretório (padrão é "C:\")	

Sessão [PPCOMP]

Aqui, configuram-se itens relacionados ao PIN-Pad compartilhado

Chave	Significado	Valor	
NaoAbrirDigitado	Configura para não abrir o digitado se cancelada a leitura do cartão no PIN-Pad compartilhado.	n	não abre digitação (padrão)
		s	abre digitação

Configuração de porta serial para Linux

Atualmente, o SCOPE Client é preparado para trabalhar apenas com PIN-Pad conectados à porta serial. No entanto, para ambiente Linux, é possível conectar um PIN-Pad serial numa porta USB, utilizando um conversor Serial←→USB. Normalmente os drivers existentes para esse conversor criam dispositivos (em /dev) nomeados **ttyUSB*x***, ao invés dos **ttyS*x***. Para o redirecionamento da porta de comunicação no SCOPE, existem duas chaves a serem colocadas no arquivo scope.ini, configuradas conforme tabela abaixo.

Chave	Significado	Valor
SerialNumPorts	Número de portas a ser configuradas.	Valor entre 1 e 6 (o padrão é 6)
SerialPort***x***	Dispositivo com novo direcionamento para a porta (*x-1*).	Valores de 0 a 5.

LEMBRETE: a configuração pode ser alternada entre as diversas portas.

Exemplo: suponhamos que possuímos 3 equipamentos seriais, mas o computador que rodará a aplicação possui uma porta serial e duas USB. Será necessário que dois equipamentos utilizem conversores Serial←→USB, que ao serem conectados, são criados dois novos dispositivos: ttyUSB0 e ttyUSB1 (os números terminais – 0 e 1 – podem variar). Portanto, em algum ponto do arquivo scope.ini, deve ser colocado a configuração abaixo para atender esta máquina:

```
SerialNumPorts=3
SerialPort0=/dev/ttyS0          ; porta 1 serial
SerialPort1=/dev/ttyUSB0        ; porta 2 USB
SerialPort2=/dev/ttyUSB1        ; porta 3 USB
```

CUIDADO: de acordo com a configuração acima, conectamos um equipamento na porta 1 e os outros dois, nas portas 2 e 3.

Sessão [PINPAD]

Aqui, configura-se item relacionado ao tamanho mínimo de dados permitidos para serem lidos pelo PIN-Pad.

Chave	Significado	Valor
TamMinDados	Tamanho mínimo de entrada de dados permitidos pelo PIN-Pad.	Valor numérico acima de 1 (o padrão é 1).

CUIDADO: Caso o PIN Pad utilize a Biblioteca Compartilhada 1.07 e PCI 2.0 a quantidade mínima de dígitos permitidos para serem lidos por coleta deve ser 4 dígitos, sendo assim, TamMinDados deve estar parametrizado com um valor de no mínimo 4.

Seção [SCOPEAPIPOS]

Essa seção permite configurações gerais, como para habilitar a digitação dos códigos e acrescentar redes e/ou bandeiras.

Chave	Significado	Valor	
DigitacaoCodigo	Configura se dá a opção de outra rede. ***ATENÇÃO: essa chave é ignorada se a chave Complementar existir.***	n	não pergunta se é outra rede
		s	pergunta se é outra e coleta o código da rede e da bandeira
Complementar	Configura se acrescenta rede à tabela padrão ou as substitui. ***ATENÇÃO: a existência dessa chave desconsidera a configuração chave DigitacaoCodigo, pois ela é prioritária.***	n	substitui as tabelas padrão pelas que estão configuradas
		s	acrescenta a configuração de redes e bandeiras que estão no arquivo a listagem padrão
Redes	Lista de até 7 códigos de redes distintas para perguntar no fluxo	Números de até 3 dígitos cada, separados por vírgula	
Bandeiras	Lista de até 14 códigos de bandeiras distintas para perguntar no fluxo	Números de até 3 dígitos cada, separados por vírgula	
NomeBandeiraYYY	Nome da bandeira YYY que será exibida no fluxo, onde YYY é uma das bandeiras listadas na chave Bandeiras. ATENÇÃO: se faltar a chave para alguma das bandeiras, será exibido código da bandeira para o operador.	String de até 20 caracteres	

Seção [SCOPEAPIPOS-RXXX]

Essa seção permite configurações de uma das redes configuradas na seção detalhada anteriormente, onde XXX é uma das redes listas na chave Redes. Portanto, deve haver uma seção para cada rede da lista.

ATENÇÃO: se não houver a seção para alguma das redes listadas, será exibido o código da rede como opção e todas as bandeiras listadas estarão associadas à rede.

Chave	Significado	Valor
Nome	Nome da rede XXX que será exibida no fluxo	String de até 20 caracteres
Bandeiras	Lista de bandeiras associadas a esta rede. A bandeira deve existir na chave Bandeiras da seção [SCOPEAPIPOS].	Números de até 3 dígitos cada, separados por vírgula
ServicoBXXX	Lista de códigos de serviços permitidos para essa bandeira, associada a essa rede, onde XXX é o código com 3 dígitos (zeros à esquerda) de uma das bandeiras listada no campo Bandeiras. Se houver bandeiras sem configuração de serviço, serão associados todos os serviços previstos.	Números de até 3 dígitos cada, separados por vírgula

Configuração do registro do MS-Windows®

No sistema operacional MS-Windows® é possível configurar o SCOPE Client utilizando o registro do sistema operacional. Abrindo o editor de registro, as configurações do SCOPE devem estar em **HKEY_LOCAL_MACHINE\SOFTWARE\Scope**:

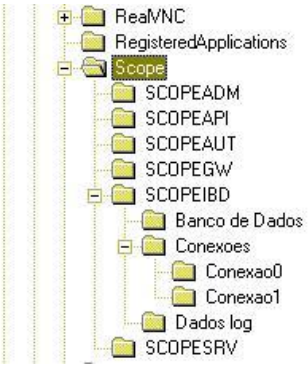


Figura 3: árvore do SCOPE no registro do MS-Windows®

Em sistemas operacionais MS-Windows® de 64 bits (x64), o SCOPE é instalado sobre o subsistema Wow64 para funcionar corretamente. No WoW64 as configurações do SCOPE são armazenadas em **HKEY_LOCAL_MACHINE/SOFTWARE/Wow6432Node/Scope**:

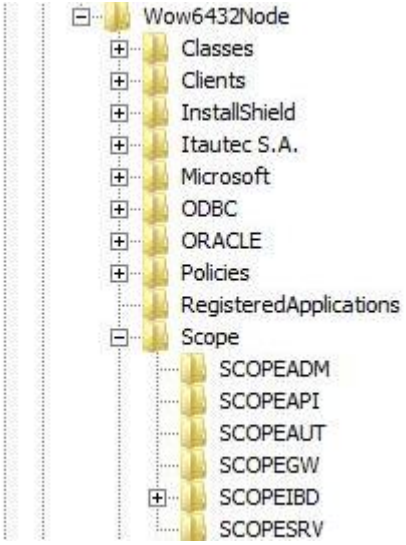


Figura 4: árvore do SCOPE no registro do Wow64®

Para maiores informações sobre o Wow64, consulte a sessão "***Instalação do SCOPE em Sistemas Operacionais Microsoft Windows® de 64-Bits***", no documento "Scope – Manual de Instalação e Configuração".

Relacionado na tabela abaixo, estão as chaves do scope.ini com o registro do MS-Windows®.

Chave do arquivo scope.ini	Localização no editor do registro
AtualizaDataHora	HKEY_LOCAL_MACHINE\SOFTWARE\Scope\SCOPEAPI\AtualizaDataHora
CupomReduzido	HKEY_LOCAL_MACHINE\SOFTWARE\Scope\SCOPEAPI\CupomReduzido

MsgOperReduzida	HKEY_LOCAL_MACHINE\SOFTWARE\Scope\SCOPEAPI\MsgOperReduzida
Name	HKEY_LOCAL_MACHINE\SOFTWARE\Scope\SCOPESRV\Name
NaoAbrirDigitado	HKEY_LOCAL_MACHINE\SOFTWARE\Scope\SCOPEAPI\NaoAbrirDigitado
Port	HKEY_LOCAL_MACHINE\SOFTWARE\Scope\SCOPESRV\Port
ShowCupom	HKEY_LOCAL_MACHINE\SOFTWARE\Scope\SCOPEAPI\ShowCupom
TimeOutAdm	HKEY_LOCAL_MACHINE\SOFTWARE\Scope\SCOPEAPI\TimeOutAdm
TimeOutLogon	HKEY_LOCAL_MACHINE\SOFTWARE\Scope\SCOPEAPI\ TimeOutLogon
VersaoAutomacao	HKEY_LOCAL_MACHINE\SOFTWARE\Scope\SCOPEAPI\VersaoAutomacao
ScopeValidaSaque	HKEY_LOCAL_MACHINE\SOFTWARE\Scope\SCOPEAPI\ScopeValidaSaque

Fontes de Dados do SCOPE no WoW64

No WoW64,as fontes de dados (ou DSNs) são criadas por um Administrador de Fontes de Dados ODBC compatível com 32 bits do WoW64 e não o de 64 bits que é acessível por padrão a partir do Painel de Controle do Windows. Assim, ao instalar o Scope são criados links com as bases de dados que podem depois ser alterados se necessário em **%SystemRoot%\SysWOW64\odbcad32.exe**.

Para maiores informações sobre o Wow64, consulte a sessão "***Instalação do SCOPE em Sistemas Operacionais Microsoft Windows® de 64-Bits***", no documento "Scope – Manual de Instalação e Configuração"

Comunicação com o ScopeSRV

O funcionamento correto da maior parte das funções do SCOPE Client demanda a conexão com o ScopeSRV. Esta conexão possui a finalidade de carregar inicialmente todos os parâmetros de configuração da aplicação, para que o SCOPE Client consiga realizar as transações básicas. Toda mensagem trocada entre o SCOPE Client e o ScopeSRV deve utilizar o protocolo TCP/IP.

Conexão

Inicia o SCOPE Client, procedendo à alocação dos recursos necessários. Essa função deverá ser executada uma única vez no início do aplicativo de PDV.

Antes de enviar o logon para o SCOPE Server, o client gera um par de chaves pública e privada. Dependendo da configuração de hardware do PDV, esse processo pode levar alguns segundos. Isso é necessário para gerar uma chave de tamanho seguro e assegurar a conformidade com o PCI-DSS. Entretanto, o estabelecimento pode definir um tempo máximo de para gerar essas chaves, configurando o parâmetro `GenTimeout` no arquivo scope.ini do client (ver tópico *Sessão [<empresa><filial>]*).

ATENÇÃO: a configuração do parâmetro `GenTimeout` poderá acarretar em não conformidade com o PCI-DSS.

Protótipo

```
LONG EXPORT ScopeOpen (char *Modo, char *Empresa, char *Filial, char *PDV)
```

Parâmetros

[in]	String (constante igual à “2”)	Modo	Modo de operação
[in]	String com quatro dígitos	Empresa	Código de identificação da empresa conforme cadastrado no ScopeCNF
[in]	String com quatro dígitos	Filial	Código de identificação da filial conforme cadastrado no ScopeCNF
[in]	String com três dígitos	PDV	Número do PDV conforme cadastrado no ScopeCNF

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA03	64003	Parâmetro 3 inválido
0xFE02	65026	SCOPE API já foi inicializada corretamente
0xFE09	65033	Servidor não configurado no arquivo scope.ini

Exemplo

```
...
char empresa[] = "0001";
char filial[] = "0001";
char pdv[] = "0001";
...
retorno = ScopeOpen ("2", empresa, filial, pdv);
if (retorno != 0)
{
   printf("Erro ao conectar com o SCOPE Server. Erro(%d)", retorno);
   exit(0);
}
...
```

Desconexão

Encerra o SCOPE Client, procedendo à liberação de todos os recursos alocados. Essa função deverá ser executada uma única vez no término do aplicativo usuário.

Protótipo

```
LONG EXPORT ScopeClose (void)
```

Parâmetros

Não há parâmetro.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE00	65024	A transação em andamento – a aplicação deve aguardar

Exemplo

```
...
retorno = ScopeClose ();
...
```

Sessão de transação

Sessão de TEF

O conceito de sessão de TEF existe para garantir a integridade de uma operação. Desta forma, entre a abertura e encerramento da sessão, todas as atividades incluídas na respectiva operação (autorização da transação, impressão de comprovante) deverão ser completamente realizadas. Este mecanismo permite que em caso de falha (queda de energia), a operação possa ser desfeita. Isso corresponde a imprimir corretamente o cupom de TEF e garantir que a falha ocorrida não atrapalhe no funcionamento da aplicação.

Sendo multi-TEF, o SCOPE implementa o conceito de sessão de TEF com o intuito de garantir que todas as transações numa mesma sessão serão aprovadas ou desfeitas. Com isto, o cliente pode efetuar o pagamento parcialmente de diversas maneiras (exemplo: 40% do valor da venda serão pagos em débito e o restante com o cartão de crédito), e para validar a venda, todas as transações deverão ser aprovadas, caso contrário, as transações não podem ser concluídas (não tem sentido que a venda seja considerada concluída apenas com a aprovação de parte do valor da compra).

Em cada venda realizada, a aplicação deve abrir a sessão, realizar as diversas transações e finalmente fechar a sessão, confirmando ou desfazendo todas as transações desta venda. A estrutura do fluxo básico de funcionamento do SCOPE Client encontra-se na figura 4.

IMPORTANTE: numa sessão de multi-TEF em que há várias transações, não é possível desfazer uma ou outra transação. Para isso, o operador deverá cancelar todas daquela sessão ou confirmá-las.

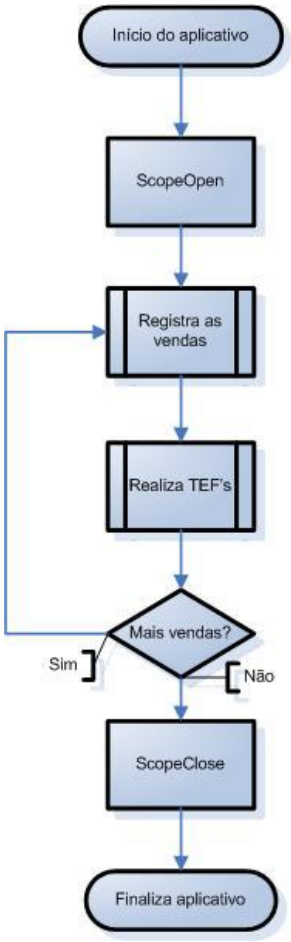


Figura 5: visão geral do funcionamento do SCOPE Client.

Abrindo uma sessão

A função ScopeAbreSessaoTEF() informa ao SCOPE para iniciar uma sessão de TEF (ciclo com uma ou mais transações TEF). Ela deve ser invocada ao finalizar a venda e antes de selecionar o meio de pagamento.

Protótipo

```
LONG EXPORT ScopeAbreSessaoTEF (void)
```

Parâmetros

Não há parâmetro.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE00	65024	A transação em andamento – a aplicação deve aguardar

Exemplo

```
...
// Conecta ao servidor do SCOPE
```

```
...
// Realiza a venda
...
retorno = ScopeAbreSessaoTEF();
if (retorno != 0)
{
   // Trata o erro e interrompe a sequência.
}
else
{
   // Inicia a(s) transação(s) desejada(s)
}
...
```

Encerrando a sessão

Aciona o SCOPE para finalizar uma sessão de TEF (ciclo com uma ou mais transações de TEF), ou seja, confirmar ou desfazer as transações da sessão em aberto, após encerrar o processamento da transação.

Protótipo

```
LONG EXPORT ScopeFechaSessaoTEF(BYTE Acao, BYTE *DesfezTEFAposQueda)
```

Parâmetros

[in]	Byte (0:Desfaz; 1:Confirma)	Acao	Informa o SCOPE para confirmar ou desfazer a(s) transação(s) da sessão de TEF atual
[out]	Ponteiro para byte	DesfezTEFAposQueda	Retorna se a(s) transação(s) da sessão de TEF foram desfeitas após uma queda de energia.

Retorno

Caso retorne sucesso (0x0000), significa que o SCOPE conseguiu com êxito confirmar ou desfazer a(s) transação(s) de uma sessão de TEF. Caso contrário, ocorreu algum problema na confirmação ou desfazimento da(s) transação(s). Para maiores detalhes dos códigos relacionados, ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFB08	64264	Erro no arquivo de controle utilizado finalização no ciclo multi-TEF
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente

Exemplo

```
BYTE acao, defez;
...
// Realiza a(s) transação(s) desejada(s)
...
```

```
acao = 1;     // confirmar
retorno = ScopeFechaSessaoTEF(acao, &defez);
if (retorno != 0)
{
   // Erro no encerramento da sessão
}
...
```

Tratando queda de energia

A função que encerra a sessão também é utilizada para desfazer as transações pendentes de uma sessão de TEF após uma queda de energia no PDV. Se o parâmetro que permite que a aplicação interfira na conclusão de transações em caso de quedas de energia estiver habilitado (ver tabela de configurações gerais), é realizado o desfazimento ou a confirmação das transações pendentes, de acordo com o parâmetro de entrada escolhido pelo operador. Se o parâmetro estiver desabilitado, após a aplicação ter restabelecido a conexão com o servidor SCOPE, ela deverá chamar essa função para finalizar uma possível sessão que ainda esteja em aberto. Nesse caso, é realizado o desfazimento das transações pendentes, independentemente, do parâmetro de entrada.

Um detalhe importante é que uma sessão de TEF é considerada finalizada ao iniciar a execução dessa função. Desta forma, mesmo que ocorra queda de energia durante a execução dessa função, o SCOPE considera como finalizado a sessão de TEF e procederá para realizar a ação solicitada.

O segundo parâmetro faz o papel principal no tratamento em queda de energia. Este parâmetro é independente do primeiro parâmetro e somente tem funcionalidade nesta situação. Caso este parâmetro seja devolvido com o valor 1 (True), a aplicação deve exibir a mensagem **"A transação TEF anterior foi desfeita (cancelada). Reter o cupom TEF"**, e se possível, acionar um supervisor para verificar a situação.

Exemplo

```
...
BYTE acao, defez;
...
// Conecta ao servidor do SCOPE
...
ScopeFechaSessaoTEF(acao, &defez);
if (desfez)
{
   printf("A transação TEF anterior foi desfeita (cancelada).");
   printf("\n Reter o cupom TEF.");
}
...
```

IMPORTANTE: sempre que tratar uma possível queda de energia e o segundo parâmetro da função ScopeFechaSessaoTEF() retornar o valor 1, deve-se imprimir a mensagem:

"A transação TEF anterior foi desfeita (cancelada). Reter o cupom TEF"

Deixando transação pendente na queda de energia

É possível também deixar a transação pendente para depois realizar o acerto de pendência manual no módulo de pendência SCOPEPND. Este acerto só deverá ser realizado por um gerente, alguém de finanças ou quem a empresa deposite confiança e seja responsável para tal procedimento.

O aplicativo deve chamar a função ScopeMTEFOnOff(), antes do tratamento de queda de energia, informando se deseja desfazer ou deixar pendentes as transações.

Protótipo

```
LONG EXPORT ScopeMTEFOnOff (BYTE Trata)
```

Parâmetros

[in]	Byte	Trata	Informa o SCOPE para desfazer (0) possíveis transações interrompidas ou deixar pendente (1) para o posterior acerto manual.

Retorno

- 0x0000 – definiu com êxito a ação a ser tomada conforme o parâmetro passado.
- 0xFE00 – não foi aberta a conexão com o SCOPE Server.
- 0xFE01 – tem transação em sendo processado.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente

Exemplo

```
...
BYTE acao, defez;
...
// Conecta ao servidor do SCOPE
...
ScopeMTEFOnOff(1);    // deixa pendente
ScopeFechaSessaoTEF(acao, &defez);
if (desfez)
{
   printf("A transação TEF anterior foi desfeita (cancelada).");
   printf("\n Reter o cupom TEF.");
}
...
```

Status de transação

Durante o processamento da transação, a aplicação pode consultar o estado em que a transação está e tomar alguma ação baseado no valor dela. Esta ação pode estar entre uma das situações abaixo, conforme o intervalo de códigos retornado pelo SCOPE Client:

- valor igual 0: transação finalizada com sucesso.
- valores entre 3 (3_h) e 92 ($5B_h$): erro de algum parâmetro passado para o SCOPE;
- valores entre 64001 ($FA01_h$) e 64005 ($FA05_h$): erro de algum parâmetro passado para o SCOPE;
- valores entre 64257 ($FB01_h$) e 64267 ($FB0B_h$): erro interno do SCOPE;
- valores entre 64512 ($FC00_h$) e 64767 ($FCFF_h$): coleta de algum dado necessário à transação;
- valores entre 65024 ($FE00_h$) e 65535 ($FFFF_h$): erro reportado pelo SCOPE referente à transação;

Enquanto o SCOPE Client devolver o código de status igual a 65024 ($FE00_h$), a aplicação deverá aguardar algum tempo até que outro status seja fornecido.

Consultando o status

A consulta do status da transação deve ocorrer enquanto uma transação (cartão de crédito, cartão de débito, recarga de celular, etc.) está em processamento.

Protótipo

```
LONG EXPORT ScopeStatus (void)
```

Parâmetros

Não há parâmetro.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente

Exemplo

```
...
// Abre sessão
retorno = ScopeCompraCartaoCredito(valor, taxa_servico);
if (retorno == 0)
{
   do
   {
      retorno = ScopeStatus();
      // trata conforme retorno
      if (retorno == 64024)
```

```
      {
         sleep (1000);
         continue;
      }
      else
         if ((retorno <= 0xFC00) || (retorno >= 0xFCFF))
            processando == 0; //saiu da faixa de coleta
      // trata a coleta de algum dado
   } while (processando == 1);
} // fim do if
else
{
   // Trata erro retornado na solicitação da transação
}
...
```

Funções de configuração de ambiente

O SCOPE Client carrega diversos parâmetros de configuração para seu funcionamento no ambiente de PDV em que está executando. Estes dados são carregados na conexão com o ScopeSRV e na leitura do arquivo scope.ini.

Além destas duas fontes de configuração, há também aquela que a aplicação pode fazer por meio de algumas funções disponíveis nas bibliotecas do SCOPE Client. Dependendo do tipo de configuração a que nos referimos, ela permite a aplicação ativar ou desativar as configurações durante a execução do programa.

Configurações gerais

No momento de execução do aplicativo, a aplicação de frente de loja poderá configurar os seguintes itens:

Código de identificação	Descrição
1	Permite que a aplicação cancele a transação que está sendo executada no PIN-Pad quando este esteja coletando algo. Por padrão, a aplicação não tem conhecimento do momento em que o SCOPE está coletando o cartão, a senha ou outra informação no PIN-Pad, pois ele retorna o estado indicando processamento (65024), o qual não é um estado de coleta, e, portanto, inviabilizando o uso da função ScopeResumeParm() para o possível cancelamento.
2	Habilitado este item, o SCOPE retornará o estado para obter os serviços (64644) durante o fluxo de TEF para que a aplicação de PDV possa obter os serviços e parâmetros configurados no ScopeCNF.
4	Se habilitado, não abrirá o digitado na leitura do cartão com o PIN-Pad Compartilhado. Por padrão, ao pressionar a tecla cancela no PIN-Pad, o SCOPE Client retornará um estado para coletar o PAN do cartão.
8	Normalmente, ao solicitar a digitação da senha no PIN-Pad pela função ScopePPStartGetPIN(), a função ScopePPGetPIN() retorna para a aplicação a senha digitada no PIN-Pad descriptografada. Habilitando este item, a senha retornada estará criptografada pela *MasterKey* da Itautec.
16	PDVs com impressora com papel de carbono podem habilitar este item para não imprimir a informação de 1ª Via Cliente ou 2ª Via Estabelecimento nos respectivos cupons.
32	Permite a gravação em arquivo dos dados da coleta para ser recuperado em caso de queda de energia.
128	Permite que a aplicação interfira na conclusão de transações que não foram

	finalizadas apropriadamente em caso de quedas de energia, confirmando ou desfazendo as transações de acordo com a decisão do operador.
256	Se habilitado, permitirá a coleta de saque em operações de débito à vista. Atualmente só é usado pela rede autorizadora Cielo.

IMPORTANTE: este tipo de configuração só é permitido quando a aplicação está utilizando a interface coleta.

Configurando em tempo de execução

A função ScopeConfigura(), responsável para a configuração da aplicação de PDV citada acima, poderá ser executada apenas uma única vez após a inicialização do SCOPE ou antes das operações TEF.

Protótipo

```
LONG EXPORT ScopeConfigura (LONG Id, LONG Param)
```

Parâmetros

[in]	LONG	Id	Identifica o parâmetro a ser configurado conforme tabela acima
[in]	LONG	Param	Informa se deve habilitar ou desabilitar o item. Para habilitar a aplicação deverá passar o valor 1 e para desabilitar, o valor 0.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFF59	65369	Função de uso exclusivo na interface coleta

Exemplo

```
...
// conexão com o scope server bem sucedida
...
// trabalhando com impressora carbonada
retorno = ScopeConfigura(16, 1);
...
```

Configuração de PIN-Pad

Para evitar a substituição de PIN-Pad por um tipo incompatível com a configuração do ScopeCNF, há a possibilidade de informar o SCOPE o tipo do PIN-Pad que está conectado e proibir a execução de transação. A validação está baseada nas seguintes opções:

- valor 0: não possui PIN-Pad conectado;
- valor 1: utilizando PIN-Pad via biblioteca VISANET;
- valor 2: utilizando PIN-Pad com biblioteca compartilhada.

Validando a interface de PIN-Pad

A validação da interface utilizada pela aplicação de PDV versus a configurada no ScopeCNF somente ocorrerá na chamada à função ScopeOpen(). Sendo assim, a aplicação de PDV deve solicitar a validação antes da conexão com o servidor do SCOPE.

Protótipo

```
LONG EXPORT ScopeValidaInterfacePP (BYTE IntPP)
```

Parâmetros

[in]	BYTE	IntPP	informa ao SCOPE a interface de acesso ao PIN-Pad compartilhada utilizada pela Aplicação PDV.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar

Exemplo

```
...
// PIN-Pad compartilhado conectado
retorno = ScopeValidaInterfacePP(2);
// conecta ao ScopeSRV
...
```

LEMBRETE: a chamada a essa função deve ser feita antes de chamar a função ScopeOpen().

Funções específicas das interfaces

Dependendo da interface sobre a qual a aplicação de PDV foi desenvolvida, a aplicação deverá tratar funções específicas da interface escolhida.

Interface coleta

Como a aplicação será responsável pela coleta (entrada de dados do usuário), esta deve obter do SCOPE Client informações para solicitar os dados. Da mesma maneira, a aplicação deve devolver o que foi coletado para o SCOPE Client. Antes de tudo, a aplicação necessita de um meio para informar ao SCOPE Client que o modo de interação será pela interface coleta. O fluxo da aplicação, quando esta utiliza a interface coleta, está esboçado no diagrama da figura 5. Nos tópicos subsequentes explicamos os mecanismos para realizar estas operações.

Na interface coleta, o SCOPE Client retornará, através da função ScopeStatus(), um código (ver códigos na tabela Coleta de dados) sempre que necessitar de uma atuação do aplicativo, como por exemplo, coletar dados, imprimir cupons e cheques, mostrar mensagens para o operador e/ou cliente, entre outros.

Definindo a interface coleta

Antes de realizar qualquer transação, a aplicação deve definir a interface coleta como meio de interação, o que é realizado pela função ScopeSetAplColeta(). Isto deverá ser feito após a conexão do SCOPE Client com o servidor.

Protótipo

```
LONG EXPORT ScopeSetAplColeta(void)
```

Parâmetros

Não há parâmetro.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE01	65025	SCOPE API não foi inicializada corretamente

Exemplo

```
...
// Conexão ao ScopeSRV bem sucedida
...
retorno = ScopeSetAplColeta();
...
```

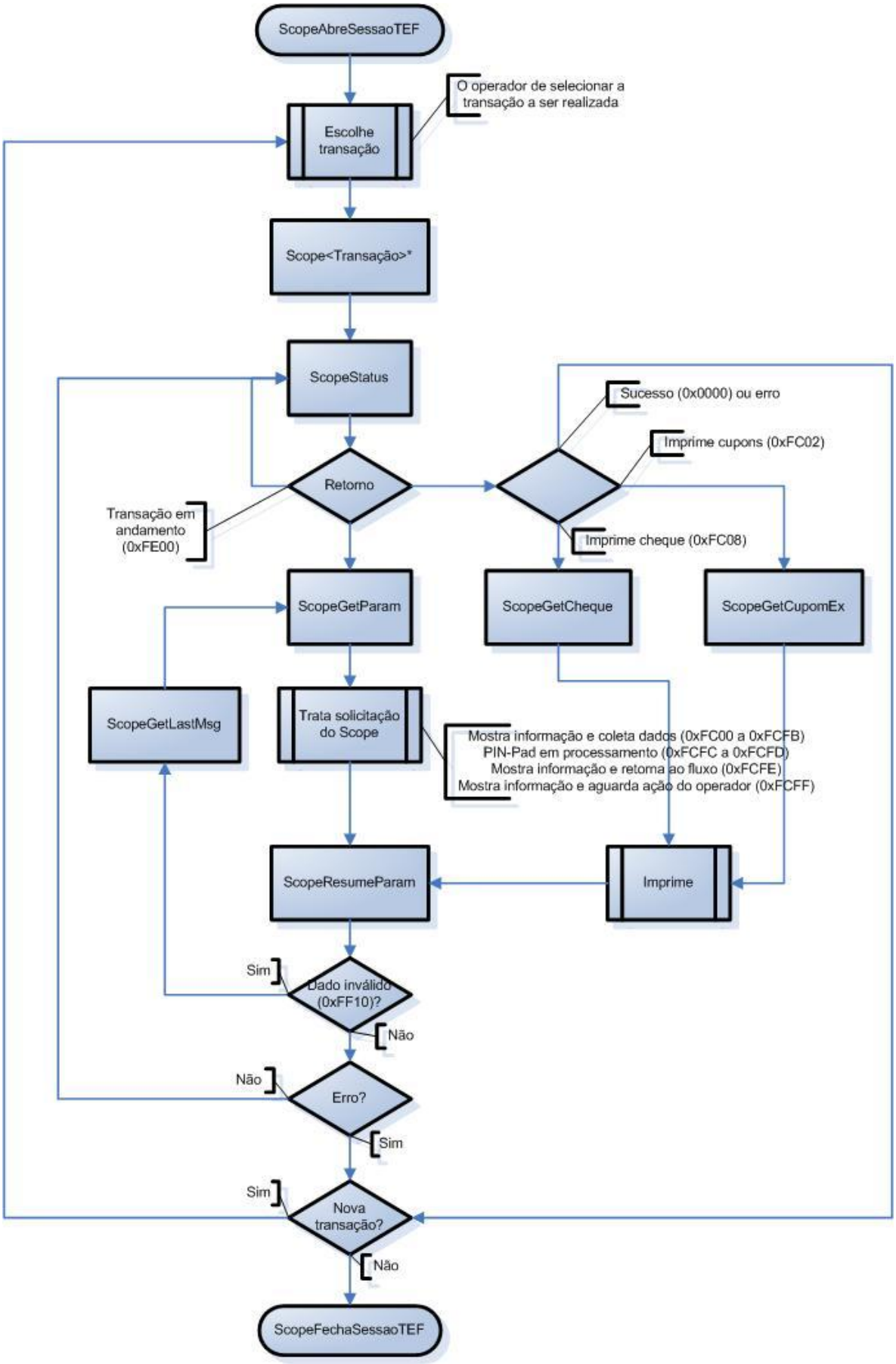

ScopeAbreSessaoTEF
O operador de selecionar a transação a ser realizada
Escolhe transação
Scope<Transação>*
ScopeStatus
Sucesso (0x0000) ou erro
Imprime cupons (0xFC02)
Retorno
Transação em andamento (0xFE00)
Imprime cheque (0xFC08)
ScopeGetParam
ScopeGetCheque
ScopeGetCupomEx
ScopeGetLastMsg
Trata solicitação do Scope
Mostra informação e coleta dados (0xFC00 a 0xFCFB)
PIN-Pad em processamento (0xFCFC a 0xFCFD)
Mostra informação e retorna ao fluxo (0xFCFE)
Mostra informação e aguarda ação do operador (0xFCFF)
ScopeResumeParam
Imprime
Sim
Dado inválido (0xFF10)?
Não
Não
Erro?
Sim
Sim
Nova transação?
Não
ScopeFechaSessaoTEF

Figura 6: fluxo de processamento de uma transação com a interface coleta.

Obtendo os parâmetros da transação

Durante a iteração do processamento de uma transação, a aplicação deve obter informações necessárias para solicitar a coleta do dado para o operador e/ou cliente. A função ScopeGetParam(), responsável em descrever os dados a coletar, deve ser chamada sempre que o SCOPE Client retornar um estado de coleta correspondente a uma coleta de dados (ver Status de transação). Com o retorno do SCOPE Client, a aplicação de PDV deverá atualizar as mensagens das telas do operador e/ou cliente, disponibilizar um meio de entrada de dado, validar a entrada, entre outras ações. A aplicação deve prover o operador de algum meio que ele possa cancelar, retornar e continuar a coleta de dados da transação.

As mensagens que o SCOPE Client disponibiliza para serem exibidas são fundamentadas no fato de que no PDV haja dois visores (display): um que é apresentado ao cliente da loja e outro, apresentado ao operador do PDV. Além disso, por padrão, considera-se que cada um desses visores possua 2 linhas de exibição de mensagens com largura de 40 caracteres. É possível configurar o tamanho máximo de colunas para até 20 ou 16 (ver o capítulo Configuração).

Estruturas de apoio

Para a aplicação receber do SCOPE Client as informações para coleta, ela deverá passar um endereço de memória de um buffer que é representado, em linguagem C, pela estrutura ptPARAM_COLETA definida no arquivo *ScopeApi.h*, cuja declaração é:

```
typedef struct _stPARAM_COLETA
{
            WORD        Bandeira;
            WORD        FormatoDado;
            WORD        HabTeclas;
            char        MsgOp1[64];
            char        MsgOp2[64];
            char        MsgCl1[64];
            char        MsgCl2[64];
            char        WrkKey[16+1];
#ifdef __linux__
            char        filler;
#endif
            WORD        PosMasterKey;
            char        PAN[19+1];
            BYTE        UsaCriptoPinpad;
            BYTE        IdModoPagto;
            BYTE        AceitaCartaoDigitado;
            char        Reservado[105];
} stPARAM_COLETA, *ptPARAM_COLETA;
```

A descrição de cada campo encontra-se abaixo:

- Bandeira: a bandeira do cartão relativa à operação em andamento (ver Código das bandeiras);
- FormatoDado: código respectivo ao formato do dado a ser coletado (ver tabela de códigos em Formato dos dados);
- HabTeclas: teclas que estão disponíveis para a aplicação. É um campo com combinação de bits. Exemplo: se o campo estiver com o valor 6 (2 + 4 = 6), significa que as teclas para prosseguir (valor 2) e retornar (valor 4) a coleta devem estar

habilitadas, desabilitando a tecla cancelar (valor 1) (ver códigos relacionados na tabela Código das teclas);

- MsgOp1: mensagem para exibição na linha 1 do visor do operador (string);
- MsgOp2: mensagem para exibição na linha 2 do visor do operador (string);
- MsgCl1: mensagem para exibição na linha 1 do visor do cliente (string);
- MsgCl2: mensagem para exibição na linha 2 do visor do cliente (string);
- WrkKey: chave de trabalho utilizada pela função PP_iGetPIN() que é responsável pela coleta de senha criptografada no PIN-Pad (string). Este campo é utilizado apenas com a biblioteca da VISANET para PIN-Pad;
- filler: para alinhamento em 8 bytes nos ambientes executados sobre o sistema operacional Linux;
- PosMasterKey: posição da *Master Key* utilizada pela função PP_iGetPIN(). Este campo é utilizado apenas com a biblioteca da VISANET para PIN-Pad;
- PAN: número do cartão utilizado pela função PP_iGetPIN(). Este campo é utilizado apenas com a biblioteca da VISANET para PIN-Pad;
- UsaCriptoPinpad: indica se a aplicação deverá coletar senha com criptografia no PIN-Pad (função PP_iGetPIN()) ou sem criptografia no PIN-Pad (função PP_iGetString()). Para o valor 1 (um), deve coletar com criptografia no PIN-Pad e para o valor 0 (zero), sem criptografia. Este campo é utilizado apenas com a biblioteca da VISANET para PIN-Pad;
- IdModoPagto: modalidade de pagamento (crédito, débito ou outros). Parâmetro utilizado na chamada da função PP_iGetCard(). Este campo é utilizado apenas com a biblioteca da VISANET para PIN-Pad;
- AceitaCartaoDigitado: indica se aceita a entrada digitada do número do cartão. Informação disponível no estado de coleta do cartão;
- Reservado: utilização futura.

Protótipo

```
LONG EXPORT ScopeGetParam (LONG tipoParam, ptPARAM_COLETA lpParam)
```

Parâmetros

[in]	LONG	TipoParam	Aplicação informa que tipo de parâmetro (retornado pelo ScopeStatus()) deseja-se obter, isto é, os parâmetros para uma coleta de dados e/ou exibição de uma mensagem.
[out]	ptPARAM_COLETA	lpParam	Conjunto de dados para a coleta detalhado acima

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno	Significado

Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFE01	65025	SCOPE API não foi inicializada corretamente

Exemplo

```
...
// executa transação (crédito, débito, etc.)
stPARAM_COLETA Pcoleta;
...
// inicia iteração
retorno = ScopeStatus();
if(retorno != 0xFE00)
{
   ...
   memset(&PColeta, '\0', sizeof(PColeta));
   retorno = ScopeGetParam(retorno, &PColeta);
   // atualiza as teclas retornar, avançar e cancelar
   // coleta os dados
   ...
   // entrega o que coletou ao SCOPE
   ...
}
// finaliza iteração
```

Passando o dado da coleta ao SCOPE Client

Realizando a coleta de dados solicitada pelo SCOPE via função ScopeStatus(), a aplicação devolverá o que foi coletado e alguma ação ao SCOPE. A ação se refere à interrupção do processamento, retorno para um estado anterior ou continuação para o próximo estado de coleta. Os dados coletados são entregues ao SCOPE por meio da função ScopeResumeParam().

Protótipo

```
LONG EXPORT ScopeResumeParam (LONG codTipoColeta, char *dados, WORD dadosParam, eACAO_APL acao)
```

Parâmetros

[in]	LONG	codTipoColeta	Código do estado, obtido pela função ScopeStatus(), para qual o dado foi coletado
[in]	String	dados	Dado coletado pela aplicação
[in]	WORD	dadosParam	Modo de captura do dado. Informa se a captura foi feita pelo teclado (0x0004) ou pela leitura magnética (0x0020) ou pela interface que suporta tanto a leitura magnética como a leitura do Chip (0x0080) ou pela leitura de CMC7 (0x0010)
[in]	eACAO_APL	Ação	Ação tomada pelo operador (ver Códigos de Fluxo)

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE04	65028	Não existe transação suspensa
0xFF10	65296	Dado inválido

Exemplo

```
...
// executa transação (crédito, débito, etc.)
char *dados[128];
WORD modo_entrada;
...
// inicia iteração
cod_coleta = ScopeStatus();
if(cod_coleta != 0xFE00)
{
   ...
   // obtém os parâmetros para a coleta
   // atualiza as teclas retornar, avançar e cancelar
   // coleta os dados e armazena em na variável 'dados'
   modo_entrada = 0x0004;   // pelo teclado
   // obtém a ação que o usuário escolheu e coloca em 'acao'
   ...
   // entrega o que coletou ao SCOPE na função abaixo
   ScopeResumeParam(cod_coleta, dados, modo_entrada, acao);
   ...
}
// finaliza iteração
```

Para resolver uma necessidade de alguns clientes que desejam manipular os itens do "Menu Dinâmico" montados através da Inicialização de Tabelas pelas redes VISANET-4.1 e CIELO, foram definidas duas novas funções exportadas pelo SCOPE Client, conforme abaixo. A ideia é que, ao receber o estado TC_EXIBE_MENU (que já existe hoje), o PDV possa opcionalmente chamar a função ScopeMenuRecuperaItens do SCOPE para receber os itens do Menu a serem exibidos e em seguida chamar a ScopeMenuSelecionaItem para indicar ao SCOPE qual item foi selecionado.

Desta forma, o PDV poderá, por exemplo:

- Exibir todos os itens na mesma tela (hoje, pelo fluxo do SCOPE Client, é apresentado um item por vez);
- Excluir algum item da lista que eventualmente não deva ser exibido ao operador;
- Alterar a descrição de algum item para melhor entendimento do operador;
- Fazer a seleção automática de um determinado item sem apresentá-lo ao operador;

OBSERVAÇÃO: O índice do item selecionado deve ser equivalente ao item da tabela fornecida pelo Scope, independente de ter sido exibido para o usuário ou não. Por exemplo, se o menu tem 10 itens, alguns deles não foram exibidos e o operador seleciona o último item do menu, o item a ser selecionado continua sendo o índice 10 (valor a ser passado pelo ScopeMenuSelecionaItem).

Protótipo

```
dllScopeAPI ScopeMenuRecuperaItens(BYTE _TipoTabela, char *_Buffer, WORD _TamBuffer);
```

Parâmetros

[in]	BYTE	_TipoTabela	Fixo em 1. Para permitir evoluções futuras.
[in]	String	_Buffer	Ponteiro para área alocada pela aplicação, para receber os itens do menu, formato abaixo.
[in]	WORD	_TamBuffer	Tamanho da área alocada disponível;

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFA03	64003	Parâmetro 3 inválido
0xFF6B	65387	Não Encontrado (para quando não veio menu dinâmico na carga de tabelas).

Estruturas do Menu Dinâmico

```
typedef struct {
     char       CodFuncao[4+1];
     char     Descricao[40+1];
     char     CodGrupoServico[2+1];
     char     CodFluxoPDV[3+1];
     char     CodRede[3+1];
     char     CodBandeira[3+1];
     char     CodFuncaoRede[4+1];
} stScopeItemMenuDinamico, *LPScopeItemMenuDinamico;

typedef struct {
     char     TipoTabela;
     char     QtdItens;
     stScopeItemMenuDinamico sItem[15];
} stScopeMenuDinamico, *LPScopeMenuDinamico;
```

Protótipo

```
dllScopeAPI ScopeMenuSelecionaItem(char _Item);
```

Parâmetros

[in]	BYTE	_Item	Índice do item selecionado

Retorno

Ver tabela de código de retorno.

O uso destas novas funções é opcional. Caso não utilizadas o tratamento do Menu Dinâmico será da mesma forma sem a chamada (todos os itens exibidos).

As configurações relativas ao Menu Dinâmico existentes hoje no SCOPE.INI do ScopeSRV serão mantidas.

Caso seja necessário alterar algum item ou comportamento do menu, recomendamos antes entrar em contato com o representante da Cielo para solicitar aprovação das alterações e verificar a necessidade de recertificação do software de frente de loja.

Interface HLAPI

Sendo o SCOPE responsável pela coleta de dados, a aplicação possui menos trabalho durante a transação. Entretanto ainda são necessários 2 outros tratamentos: obtenção de dados para uso da aplicação e notificação do SCOPE para impressão do cupom (figura 6).

Muitas vezes a aplicação necessita controlar a venda e seu meio de pagamento. Para aquelas vendas que são pagas com cartão, a aplicação pode manter na informação da venda, alguns dados da TEF para controle de caixa. As informações da TEF que a aplicação poderá precisar para esse controle são várias. O SCOPE Client fornece a função ScopeObtemCampoExt() para a obtenção dos dados da transação.

A impressão de cupons é outro ponto de atenção que a aplicação deve tratar para compensar este tipo de interface. Uma vez que na interface HLAPI o SCOPE Client coleta todos os dados e a aplicação não têm conhecimento do que ele está fazendo, num determinado momento ela deve pedir para o SCOPE suspender a transação, para que o cupom seja impresso. Uma vez suspensa, a aplicação obtém os cupons, imprime-os e informa ao SCOPE que ele poderá continuar ou até mesmo cancelar a transação.

Suspendendo a transação

A aplicação deve fazer isso ao menos uma vez na interface HLAPI antes de cada transação. A aplicação poderá solicitar ao SCOPE Client para que ele suspenda em diversos momentos. Mas, ao menos uma vez, porque é primordial que a transação esteja suspensa para imprimir o cupom. Existem diversos códigos representando pontos ou estados de coleta em que o SCOPE Client pode suspender a transação, sendo feitos com a função ScopeSuspend().

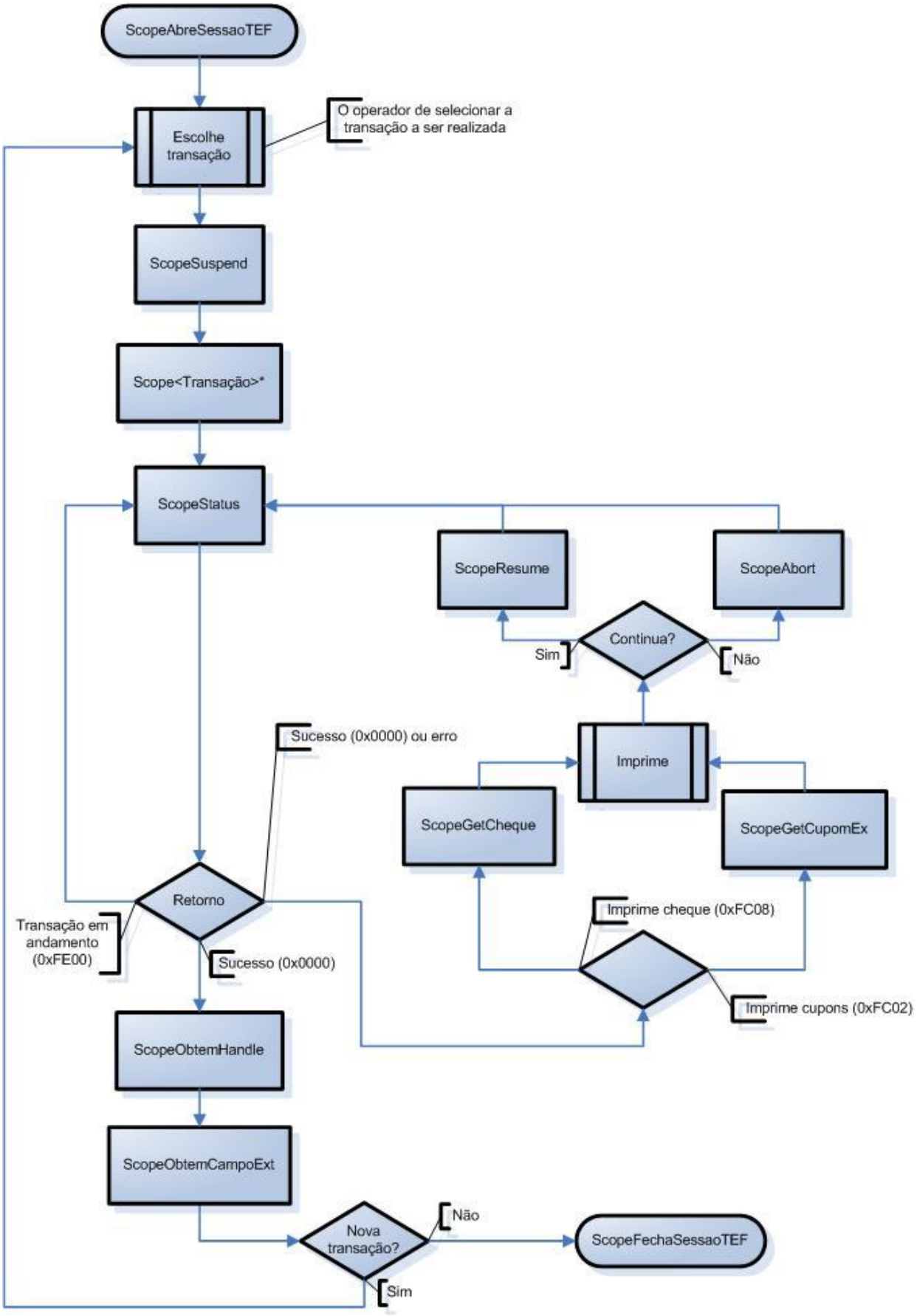


Figura 7: fluxo de processamento de uma transação com a interface HLAPI.

Protótipo

```
LONG EXPORT ScopeSuspend (LONG estado)
```

Parâmetros

[in]	LONG	estado	Estado de parada. Indica o passo de execução após o qual o controle de execução retornará ao aplicativo. O SCOPE Client manterá o contexto de saída e aguardará que o aplicativo retome, ou aborte o seu fluxo de execução. (Ver a lista de códigos de estado na tabela Estados para interrupção)

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFE01	65025	SCOPE API não foi inicializada corretamente

Exemplo

```
...
// definindo o ponto de suspensão
retorno = ScopeSuspend(4);
...
```

LEMBRETE: uma vez suspensa a transação, o SCOPE Client não retomará sozinho. É necessário que a aplicação notifique o SCOPE para retoma-la.

Retomando a transação

Todo o tempo em que a transação está suspensa, a aplicação fará algo que ela necessita. Realizado o que a aplicação deveria fazer enquanto a transação está suspensa, a aplicação deve informar para o SCOPE que pode continuar, caso contrário, a transação não será concluída.

Protótipo

```
LONG EXPORT ScopeResume (void)
```

Parâmetros

Não há parâmetro.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno	Significado

Hexadecimal	Decimal	
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE04	65028	Não existe transação suspensa

Exemplo

```
...
// SCOPE passou o controle para a aplicação, suspendendo a transação
...
// A aplicação fez o que desejava (ex: imprimiu o cupom)
...
retorno = ScopeResume();
...
```

Abortando a transação

Em certos momentos, a transação não pode ser concluída. Por problema na impressora, desistência do cliente, etc. Enquanto a transação estiver suspensa, caso se deseje cancelar a transação, a aplicação poderá continuar a transação e fechar a sessão solicitando que o SCOPE desfaça-a ou simplesmente aborte no ponto em que está suspensa. A complicação em desfazer a transação no momento do encerramento da sessão, é que todas as transações serão desfeitas. A função ScopeAbort() aborta o fluxo de execução do SCOPE Client suspenso pela função ScopeSuspend() ou quando o SCOPE Client está aguardando uma ação do aplicativo.

Protótipo

```
LONG EXPORT ScopeAbort(void)
```

Parâmetros

Não há parâmetro.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE04	65028	Não existe transação suspensa
0xFB01	64257	Não foi possível criar a "*thread*" na coleta de dados
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF

Exemplo

```
...
// transação suspensa
...
// deseja-se cancelar a transação
ScopeAbort();
...
```

CUIDADO: só é possível abortar uma transação quando ela está suspensa.

Nas seções seguintes, trataremos sobre as transações com cartão crédito e débito, desde compras a saldo até as próprias consultas de saldo.

Cartão de crédito

Uma das operações mais realizadas numa loja e muito incentivada pelas empresas de cartões, o pagamento de compra com o cartão de crédito apresenta diversos serviços relacionados: compra à vista, parcelada pela loja, parcelada pela administradora de cartão, consulta de saldo e de financiamento e cancelamento (que dedicamos um capítulo exclusivo).

Compra com cartão de crédito

Embora haja diversos serviços relacionados ao cartão de crédito, apenas uma função está disponível: ScopeCompraCartaoCredito(). Ela se refere ao grupo de serviço de crédito e trata os seus serviços. O que determinará cada um destes serviços será o fluxo do SCOPE de acordo com o que este solicitar e o que o usuário devolver. Assim, a transação poderá terminar como à vista, parcelada pelo estabelecimento ou parcelada pela administradora.

Protótipo

```
LONG EXPORT ScopeCompraCartaoCredito (char *Valor, char *TxServico)
```

Parâmetros

[in]	String com o máximo de 12 dígitos	Valor	Valor da transação com a vírgula implícita (exemplo: R$ 123,00 = “12300”)
[in]	String	TxServico	Valor da taxa de serviço

Retorno

Ver tabela de código de retorno.

Estados de coleta

A tabela abaixo mostra os estados de coleta que o fluxo poderá retornar, no modo coleta.

Códigos Retorno		Significado
Hexadecimal	**Decimal**	
0xFC00	64512	Coletar cartão
0xFC01	64513	Coletar validade do cartão
0xFC02	64514	Imprime Cupom
0xFC03	64515	Coletar CGC ou CPF
0xFC09	64521	Coletar se a transação será a vista ou não
0xFC0E	64526	Coletar quantidade de parcelas
0xFC0A	64522	Coletar se a transação será parcelada pela administradora ou pelo estabelecimento
0xFC18	64536	Coleta os 4 últimos dígitos do cartão
0xFC1F	64543	Coletar quantidade de dias
0xFC24	64548	Coletar número do endereço

0xFC26	64550	Coletar plano de pagamento (Fininvest)
0xFC28	64552	Coletar número do item (Fininvest)
0xFC34	64564	Coleta o valor da transação
0xFC53	64595	Coleta código do material
0xFC6D	64621	Coleta a data de emissão do cartão
0xFC6E	64622	Coleta o plano Infocards
0xFC6F	64623	Coleta número do cupom fiscal
0xFC0E	64526	Coletar quantidade de parcelas
0xFC15	64533	Coletar valor de entrada
0xFC1B	64539	Imprime consulta
0xFC20	64544	Coletar o número da pré-autorização
0xFC23	64547	Coletar CEP
0xFC33	64563	Coleta valor da taxa de serviço
0xFC64	64612	Coleta RG
0xFC7E	64638	Go On Chip
0xFC80	64640	Coleta o valor da taxa de embarque
0xFC84	64644	Obtém os serviços
0xFC85	64645	Coleta o cartão digitado
0xFC8A	64650	Coleta a data quando o cliente aderiu ao cartão
0xFC29	64553	Coletar código de segurança
0xFC2F	64559	Coleta código da localidade do telefone
0xFC30	64560	Coleta número do telefone
0xFC51	64593	Imprime cupom promocional
0xFCA5	64677	Coleta o número do Voucher
0xFCFE	64766	Mostrar informações e retornar fluxo para o cliente SCOPE
0xFCFF	64767	Mostrar Informações e aguardar confirmação do operador

Possíveis Retornos de Erros

Códigos Retorno		**Significado**
Hexadecimal	**Decimal**	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFB01	64257	Não foi possível criar a "*thread*" na coleta de dados
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE05	65029	API ainda não fez nenhuma transação após a inicialização. A aplicação está tentando obter um *handle* sem ter feito nenhuma transação desde última conexão com o ScopeSRV
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFF0C	65292	Transação não implementada

0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFF36	65334	Erro no envio da confirmação ou desfazimento da transação anterior
0xFF3A	65338	Erro interno na execução da coleta
0xFF60	65376	Função indisponível
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
char valor[12 + 1];
char taxa[12 + 1];
...
// obtém o valor da compra e armazena em 'valor'
// obtém a taxa de serviço e armazena em 'taxa'
// abre sessão
...
retorno = ScopeCompraCartaoCredito(valor, taxa);
...
// processa a transação
...
// fecha a sessão
...
```

Consulta a financiamento de cartão de crédito

Antes de realizar uma compra financiada pela administradora de cartão de crédito, é possível consulta das parcelas do financiamento. A consulta seguirá os moldes de uma transação de compra, sendo que em nenhum momento será efetivada. Portanto, após a consulta, é necessário realizar a compra, que solicitará os mesmos dados da consulta.

Protótipo

```
LONG EXPORT ScopeConsultaCredito (char *Valor, char *TxServico)
```

Parâmetros

[in]	String com o máximo de 12 dígitos	Valor	Valor da transação com a vírgula implícita (exemplo: R$ 123,00 = "12300")
[in]	String	TxServico	Valor da taxa de serviço

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado

0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
char valor[12 + 1];
char taxa[12 + 1];
...
// obtém o valor da compra e armazena em 'valor'
// obtém a taxa de serviço e armazena em 'taxa'
// abre sessão
...
retorno = ScopeConsultaCredito(valor, taxa);
...
// processa a transação
...
// fecha a sessão
...
```

Consulta a saldo de cartão de crédito

Um dos motivos que uma transação pode não ser aprovada é a falta de saldo suficiente do cliente no cartão de crédito. Alguns cartões permitem esta consulta.

Protótipo

```
LONG EXPORT ScopeConsultaSaldoCredito (void)
```

Parâmetros

Não há parâmetros.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE00	65024	A transação em andamento – a aplicação deve aguardar

0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
*0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
// abre sessão
...
retorno = ScopeConsultaSaldoCredito();
...
// processa a transação
...
// fecha a sessão
...
```

Pré-autorização de crédito

Como a consulta de saldo do cartão de crédito não está disponível para todos os cartões e por representar certo risco (cartões roubados poderão ter todo o seu limite gasto, sendo conhecido o seu saldo), outras administradoras disponibilizam a opção de pré-autorização. Assim, antes de fazer a compra, é possível realizar uma pré-autorização para verificar se a compra procederá com sucesso.

Protótipo

```
LONG EXPORT ScopePreAutorizacaoCredito (char *Valor, char *TxServico)
```

Parâmetros

[in]	String com o máximo de 12 dígitos	Valor	Valor da transação com a vírgula implícita (exemplo: R$ 123,00 = "12300")
[in]	String	TxServico	Valor da taxa de serviço

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
char valor[12 + 1];
char taxa[12 + 1];
...
// obtém o valor da compra e armazena em 'valor'
// obtém a taxa de serviço e armazena em 'taxa'
// abre sessão
...
retorno = ScopePreAutorizacaoCredito(valor, taxa);
...
// processa a transação
...
// fecha a sessão
...
```

Cartão de débito

Abrangendo diversos serviços, como o cartão de débito, é uma das transações mais realizadas nos estabelecimentos. Falando em serviços, nos referimos a débito à vista, débito pré-datado, débito parcelado com a 1ª parcela agendada ou à vista, débito Voucher, saque e CDC.

Compra com cartão de débito

Para os serviços de débito à vista, débito pré-datado, débito parcelado com a 1ª parcela agendada ou á vista e débito Voucher, deve-se utilizar a função ScopeCompraCartaoDebito(). Como as outras transações, o fluxo de processamento é similar.

Protótipo

```
LONG EXPORT ScopeCompraCartaoDebito (char *Valor)
```

Parâmetros

[in]	String com o máximo de 12 dígitos	Valor	Valor da transação com a vírgula implícita (exemplo: R$ 123,00 = “12300”)

Retorno

Ver tabela de código de retorno.

Estados de coleta

A tabela abaixo mostra os estados de coleta que o fluxo poderá retornar, no modo coleta

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFC00	64512	Coletar cartão
0xFC01	64513	Coletar validade do cartão
0xFC02	64514	Imprime Cupom
0xFC03	64515	Coletar CGC ou CPF
0xFC09	64521	Coletar se a transação será a vista ou não
0xFC0B	64523	Coletar se a transação será pré-datada
0xFC0C	64524	Coletar se a parcela será à vista
0xFC0D	64525	Coletar quantidade de dias entre parcelas
0xFC0E	64526	Coletar quantidade de parcelas
0xFC10	64528	Coletar o dia e o mês (DDMM)
0xFC11	64529	Coletar a senha
0xFC13	64531	Coletar a forma de pagamento
0xFC14	64532	Coletar data do primeiro vencimento
0xFC15	64533	Coletar valor de entrada
0xFC18	64536	Coletar últimos dígitos do cartão
0xFC1A	64538	Coletar se deseja consultar parcelas
0xFC1B	64539	Imprime consulta
0xFC22	64546	Imprime nota promissória
0xFC23	64547	Coletar CEP
0xFC24	64548	Coletar número do endereço
0xFC25	64549	Coletar parte numérica do complemento
0xFC2B	64555	Coleta se é com ou sem garantia de pré-datado
0xFC2C	64556	Coleta se aceita ou não risco
0xFC30	64560	Coleta número do telefone

0xFC32	64562	Coleta data (formato DDMMAA)
0xFC33	64563	Coleta valor da taxa de serviço
0xFC47	64583	Coleta quantidade de parcelas e aceita 1 parcela
0xFC51	64593	Imprime cupom promocional
0xFC52	64594	Coleta se utiliza saldo
0xFC64	64612	Coleta RG
0xFC66	64614	Coleta somente CPF
0xFC7B	64635	Coleta o valor da primeira parcela
0xFC7D	64637	Coleta se cancela ou não a transação
0xFC7E	64638	Go On Chip
0xFC7F	64639	Retira o cartão
0xFC80	64640	Coleta o valor da taxa de embarque
0xFC85	64645	Coleta o cartão digitado
0xFC87	64647	Exibe o menu
0xFC95	64661	Coleta o valor das parcelas
0xFC96	64662	Coleta se a primeira parcela é para 30 ou 60 dias
0xFC9F	64671	Coleta a placa do veículo (transação de convênio combustível Banrisul)
0xFCA5	64677	Coleta o número do Voucher
0xFCBE	64702	Coleta dados ECF
0XFCC1	64705	Coleta opção de pagamento de carnê para GetnetLAC no serviço de débito a vista para bandeira ELECTRON (0=Não para pagamento de carnê, 1=Sim para pagamento de carnê).
0xFCCB	64715	*Coleta da Lista de Mercadorias Consumidas
0xFCFE	64766	Mostrar informações e retornar fluxo para o cliente SCOPE
0xFCFF	64767	Mostrar Informações e aguardar confirmação do operador

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	**Decimal**	
0xFA01	64001	Parâmetro 1 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
char valor[12 + 1];
...
// obtém o valor da compra e armazena em 'valor'
```

```
...
// abre sessão
...
retorno = ScopeCompraCartaoDebito(valor);
...
// processa a transação
...
// fecha a sessão
...
```

Compra CDC (Crédito Direto ao Consumidor)

Apesar de existir uma função ScopeCompraCDC() responsável por essa modalidade de serviço, esta transação foi incorporada ao fluxo da transação ScopeCompraCartaoDebito() devido exigência da especificação Visanet 4.1. O que difere no fluxo de um pagamento com cartão de débito é que um CDC é um débito parcelado pela administradora.

Caso no fluxo de compra CDC for realizado uma consulta, os cupons da consulta não poderão ser impressos devido a problemas de controle fiscal. Estes cupons deverão ser exibidos no visor para o cliente. Somente os cupons da compra é que deverão ser impressos caso haja interesse em continuar a compra após a consulta.

Consulta CDC

Para uma simples consulta CDC, independente da compra, deve-se utilizar a função específica para a transação: ScopeConsultaCDC(). O resultado desta transação será um cupom com valores do parcelamento com as taxas.

Quando a consulta for realizada por esta função, os cupons da consulta deverão ser enviados para a impressora.

Protótipo

```
LONG EXPORT ScopeConsultaCDC (char *Valor, char *TxServico)
```

Parâmetros

[in]	String com o máximo de 12 dígitos	Valor	Valor da transação com a vírgula implícita (exemplo: R$ 123,00 = “12300”)
[in]	String	TxServico	Valor da taxa de serviço

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server

0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFF0C	65292	Transação não implementada
0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
char valor[12 + 1];
char taxa[12 + 1];
...
// obtém o valor da compra e armazena em 'valor'
// obtém a taxa de serviço e armazena em 'taxa'
// abre sessão
...
retorno = ScopeConsultaCDC(valor, taxa);
...
// processa a transação
...
// fecha a sessão
...
```

LEMBRETE: a compra CDC é efetuada no fluxo de débito, ou seja, deverá ser chamada a função ScopeCompraCartaoDebito() para a realização dessa compra**.**

Simulação Crediário

Atualmente esta transação é de uso exclusivo da rede Cielo. Seu uso é opcional, já que a mesma transação pode ser feita através da ScopeMenu. Quando feita através desta última, segue as regras de seleção de produtos da Cielo.

Protótipo

```
LONG EXPORT ScopeSimulacaoCrediario (WORD CodGrupoServico, char *Valor,
char *TxServico)
```

Parâmetros

[in]	Numérico	CodGrupoServico	Reservado para uso futuro. Há previsão de

			esta funcionalidade ser usada através de cartões de Crédito no futuro, mas atualmente está disponível apenas para cartões de Débito. Portanto, usar fixo 1 (Débito).
[in]	String com o máximo de 12 dígitos	Valor	Valor da transação com a vírgula implícita (exemplo: R$ 123,00 = “12300”)
[in]	String	TxServico	Valor da taxa de serviço

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	**Decimal**	
0xFA01	64001	Parâmetro 1 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV’s disponíveis. Erro retornado quando a aplicação passa o caractere “#” como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV’s disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFF0C	65292	Transação não implementada
0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
char valor[12 + 1];
char taxa[12 + 1];
...
// obtém o valor da compra e armazena em ‘valor’
// obtém a taxa de serviço e armazena em ‘taxa’
// abre sessão
...
retorno = ScopeSimulacaoCrediario(1, valor, taxa);
...
// processa a transação
...
```

```
// fecha a sessão
...
```

LEMBRETE: A transação de Compra Crediário é efetuada no fluxo de débito, ou seja, deverá ser chamada a função ScopeCompraCartaoDebito() para a realização dessa compra. Quando configurada na inicialização da Cielo (tabela 7C de Produtos), o SCOPE irá exibir um menu de seleção, por exemplo (Débito ou Crediário). Se selecionado Débito, o SCOPE seguirá o fluxo convencional de Débito. Se selecionado Crediário o SCOPE seguirá o fluxo específico desta transação.

Consulta a saldo de cartão de débito

Um dos motivos que uma transação pode não ser aprovada é a falta de saldo suficiente do cliente no cartão de débito. Alguns cartões permitem esta consulta.

Protótipo

```
LONG EXPORT ScopeConsultaSaldoDebito (char *Valor)
```

Parâmetros

[in]	String com o máximo de 12 dígitos	Valor	Valor da transação com a vírgula implícita (exemplo: R$ 123,00 = “12300”)

Retorno

Ver tabela de código de retorno.

Exemplo

```
...
char valor[12 + 1];
...
// obtém o valor da compra e armazena em 'valor'
...
// abre sessão
...
retorno = ScopeConsultaSaldoDebito(valor);
...
// processa a transação
...
// fecha a sessão
...
```

Cartão Dinheiro

São cartões ao portador, protegido por senha, semelhante ao cartão de débito que é adquirido (comprado) com um valor fixo, e posteriormente pode ser recarregado. È utilizado principalmente para efetuar compras, debitando esse valor do saldo do cartão, cujo objetivo é a substituição de dinheiro por este cartão. Também, é conhecido como Cartão Presente ou Gift Card.

Operações

As operações (transações) que podem ser efetuadas com o cartão dinheiro são:

- Compra/Carga de cartão
- Compra usando cartão (débito)
- Consulta a saldo
- Estornos

Compra/Carga do cartão dinheiro

Para realizar a operação de recarga, e também, a compra (carga inicial) do cartão dinheiro, deve-se utilizar a função ScopeCartaoDinheiro(), como descrito abaixo.

Como as outras transações, o fluxo de processamento é similar.

Protótipo

```
LONG EXPORT ScopeCartaoDinheiro (WORD servico, char *Valor)
```

Parâmetros

[in]	WORD	Servico	Para efetuar a operação de Compra/Carga de cartão dinheiro usar o serviço **110 S_CARGA_CARTAO_DINHEIRO**
[in]	String com o máximo de 12 dígitos	Valor	Valor da transação com a vírgula implícita (exemplo: R$ 123,00 = “12300”)

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFF0C	65292	Transação não implementada

Exemplo

```
...
Long servico = S_CARGA_CARTAO_DINHEIRO; // valor de 110
char valor[12 + 1];
...
```

```
// obtém o valor da compra e armazena em 'valor'
// abre sessão
...
retorno = ScopeCartaoDinheiro(servico, valor);
...
// processa a transação
...
// fecha a sessão
...
```

Compra usando o cartão dinheiro (débito)

Para realizar a operação compra usando o cartão dinheiro, isto é, debitar o valor da compra do saldo disponível, pode ser feito de duas formas:

1. Usar a função ScopeCompraCartaoDebito(), como se fosse um cartão de débito normal (vide item *Compra com cartão de débito*).
2. Usar a função ScopeCartaoDinheiro(), como descrito abaixo.

Como as outras transações, o fluxo de processamento é similar.

Protótipo

```
LONG EXPORT ScopeCartaoDinheiro (WORD servico, char *Valor)
```

Parâmetros

[in]	WORD	Servico	Para efetuar a operação de Compra/Carga de cartão dinheiro usar o serviço **98 S_COMPRA_CARTAO_DINHEIRO**
[in]	String com o máximo de 12 dígitos	Valor	Valor da transação com a vírgula implícita (exemplo: R$ 123,00 = "12300")

Retorno

Ver tabela de código de retorno.

Exemplo

```
...
Long servico = S_COMPRA_CARTAO_DINHEIRO;        // valor de 98
char valor[12 + 1];
...
// obtém o valor da compra e armazena em 'valor'
// abre sessão
...
retorno = ScopeCartaoDinheiro(servico, valor);
...
// processa a transação
...
// fecha a sessão
...
```

Consulta ao saldo de cartão dinheiro

A consulta ao saldo de cartão dinheiro e a respectiva impressão de cupom deve ser feita usando a função ScopeConsultaCartaoDinheiro(), como descrito abaixo.

Protótipo

```
LONG EXPORT ScopeConsultaCartaoDinheiro (void)
```

Parâmetros

Não há parâmetros.

Retorno

Ver tabela de código de retorno.

Exemplo

```
...
// abre sessão
...
retorno = ScopeConsultaCartaoDinheiro();
...
// processa a transação
...
// fecha a sessão
...
```

Estornos

Tanto para realizar a operação de estorno de compra/carga do cartão dinheiro como a compra usando o cartão dinheiro (débito) deve se usar a função genérica de estorno chamada ScopeCancelamento(). A documentação desta função esta descrita no capítulo Estorno de transações.

Funções de Consulta

Neste capítulo falamos sobre as transações de consultas que não se referem aos cartões.

Cheque

Outra funcionalidade que o SCOPE dispõe é a consulta de cheques no Serasa ou na ACSP – Associação Comercial de São Paulo. Nas consultas de cheques, via PDVs, as respostas são baseadas no número do documento (CPF/CGC), banco, número do cheque do comprador, valor e data de vencimento da compra.

O número do cheque é composto por seis caracteres numéricos, por definição do Banco Central do Brasil com base na resolução 885, de 22/12/1983.

Consulta de cheques

A transação de consulta a cheque é iniciada pela função ScopeConsultaCheque(). O fluxo de processamento é similar às outras transações.

Protótipo

```
LONG EXPORT ScopeConsultaCheque (char *Valor)
```

Parâmetros

[in]	String com o máximo de 12 dígitos	Valor	Valor da transação com a vírgula implícita (exemplo: R$ 123,00 = “12300”)

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV’s disponíveis. Erro retornado quando a aplicação passa o caractere “#” como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV’s disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
char valor[12 + 1];
...
// obtém o valor do cheque e armazena em 'valor'
// abre sessão
...
retorno = ScopeConsultaCheque(valor);
...
// processa a transação
...
// fecha a sessão
...
```

AVS

Termo em inglês "*Address Verification Service*", também conhecido por outras autorizadoras como VAS – Verificação Automática de Endereço, é um serviço de verificação de endereço do portador do cartão utilizado para comparar o endereço que o cliente forneceu com o que está cadastrado junto com a administradora de cartão. O AVS é uma transação não financeira e não é atrelada a nenhuma transação de venda, isto é, a consulta não garante que a pessoa que esteja realizando a transação seja realmente o portador do cartão.

CUIDADO: Este tipo de informação recebida pela consulta AVS tem o intuito apenas de oferecer suporte para a decisão do estabelecimento comercial, quanto à realização ou não da venda.

Consulta AVS

A função ScopeConsultaAVS() aciona o SCOPE Client para efetuar uma transação de consulta AVS para confirmar os dados do endereço fornecido com o cadastrado na administradora.

Protótipo

```
LONG EXPORT ScopeConsultaAVS (void)
```

Parâmetros

Não há parâmetros.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar

0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFF0C	65292	Transação não implementada
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
// abre sessão
...
retorno = ScopeConsultaAVS();
...
// processa a transação
...
// fecha a sessão
...
```

Recarga de celular

Com o crescimento número de pessoas que utilizam celulares do tipo pré-pago, a necessidade de disponibilizar pontos de recarga tem sido incentivada pelas operadoras de celulares. O SCOPE contribui com este “movimento” provendo facilidades para esse tipo de serviço.

Configurando a recarga de celular

Para a Recarga de Celular devem ser efetuadas as configurações a seguir:

- ScopeCNF: ver no documento 'Scope - Manual de Instalação e Configuração.doc' cadastrando os atributos para a Recarga de Celular.
- ScopeINI: O tamanho mínimo de dígitos permitidos para serem lidos pelo PIN-Pad deve ser configurado como descrito no item "Configuração do arquivo scope.ini" seção "Sessão [PINPAD]" deste documento.

Processando a recarga de celular

Apesar de o processamento seguir de maneira similar às outras transações, nesta modalidade existem códigos de coleta específicos. Durante a iteração do processamento da transação de recarga de celular, a aplicação de PDV deverá estar preparada para receber, da função ScopeStatus(), no caso de uso da interface Coleta, os dois códigos de coleta abaixo, além de outros comuns a outras transações:

- Código 64624 (0xFC70): neste momento, a aplicação deverá recuperar do SCOPE a lista de operadoras disponíveis;
- Código 64558 (0xFC2E): este código representa que a aplicação deve obter a lista de valores disponíveis de recarga.

No caso da interface HLAPI, antes do início da transação de recarga (chamada à função ScopeRecargaCelular()), a aplicação deve informar ao SCOPE que o processamento deve ser suspenso (via ScopeSuspend()) no estado 187 (coleta a operadora). Quando a ScopeStatus() retornar 65027 (existe transação suspensa), a aplicação deve receber a lista de operadoras disponíveis (via ScopeRecuperaOperadorasRecCel()), exibi-la ao usuário de alguma forma, informar ao SCOPE um novo estado de suspensão, o 95 (coleta o valor de recarga de celular) e dar continuidade ao processamento com a ScopeResume(). Posteriormente, a ScopeStatus() irá retornar novamente o valor 65027 (existe transação suspensa), nesse contexto, indicando o momento para a exibição dos valores de recarga disponíveis. Tais valores serão obtidos pela função ScopeRecuperaValoresRecCel() e deverão ser exibidos ao usuário pela aplicação. A aplicação deve informar um novo estado de suspensão, 4 (imprime cupom) e permitir a continuidade do processamento com a ScopeResume().

Iniciando a transação de recarga

O início desta transação é dado pela chamada à função ScopeRecargaCelular(). Esta transação como todas as outras, deve estar numa sessão de TEF (ver Sessão de transação).

Protótipo

```
LONG EXPORT ScopeRecargaCelular (void)
```

Parâmetros

Não há parâmetros.

Retorno

Ver tabela de código de retorno.

Estados de coleta

A tabela abaixo mostra os estados de coleta que o fluxo poderá retornar, no modo coleta

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFC02	64514	Imprime Cupom
0xFC2E	64558	Coleta valor da recarga de celular pré-pago
0xFC2F	64559	Coleta código da localidade do telefone
0xFC30	64560	Coleta número do telefone
0xFC70	64624	Coleta a operadora de recarga de celular pré-pago
0xFC90	64656	Coleta o DDD no PINPad
0xFCB5	64693	Coleta redigitação do DDD
0xFCBC	64700	Coleta DDD + Telefone no PIN Pad
0xFCBD	64701	Redigita DDD + Telefone no PIN Pad

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
// abre sessão
...
```

```
retorno = ScopeRecargaCelular();
...
// processa a transação
...
// fecha a sessão
...
```

Obtendo operadoras disponíveis

Recebendo o código de coleta 64624 da função ScopeStatus(), a aplicação deverá obter as operadoras disponíveis, através da função ScopeRecuperaOperadorasRecCel(). A lista disponibilizada no segundo parâmetro da função contém o formato abaixo:

Posição	Formato	Descrição
01 até 02	Short	Quantidade de operadoras disponíveis
03	Byte	Código da operadora 1
04 até 25	String	Nome da operadora 1
26	Byte	Código da operadora 2
27 até 48	String	Nome da operadora 2
…	…	…
$Pos_n = (n-1)\times 23+3$	Byte	Código da operadora n
(Pos_n+1) até (Pos_n+21)	String	Nome da operadora n

Exemplo: Quando houver 5 operadoras, as posições dos códigos de cada operadora no buffer serão:

$$Pos_1 = (1-1)\times 23+3 = 3$$
$$Pos_2 = (2-1)\times 23+3 = 26$$
$$Pos_3 = (3-1)\times 23+3 = 49$$
$$Pos_4 = (4-1)\times 23+3 = 72$$
$$Pos_5 = (5-1)\times 23+3 = 95$$

Portanto, codificando de maneira simples em linguagem C, temos:

```
...
short *qtd_op;
char buffer[2002];
int operadora, ind_op;
long retorno;
...
retorno = ScopeRecuperaOperadorasRecCel (2, buffer, sizeof(buffer));
if(retorno == 0)
{
   qtd_op = (short *) buffer;
   for(operadora = 0; operadora < *qtd_op; operadora++)
   {
      ind_op = (operadora - 1) * 23 + 3;
```

```
      printf("\nCod %d  - Oper. %.21s",
                              (int) buffer[ind_op], buffer[ind_op+1]);
   }
}
...
```

No entanto, de outra maneira mais iterativa e sem utilizar fórmulas matemáticas, podemos obter a lista de operadoras como segue no exemplo abaixo:

```
...
short *qtd_op;
char buffer[2002];
int operadora, ind_op;
long retorno;
...
retorno = ScopeRecuperaOperadorasRecCel (2, buffer, sizeof(buffer));
if(retorno == 0)
{
   qtd_op = (short *) buffer;
   ind_op = 3;
   for(operadora = 0; operadora < *qtd_op; operadora++)
   {
      printf("\nCod %d  - Oper. %.21s",
                              (int) buffer[ind_op], buffer[ind_op+1]);
      ind_op = ind_op + 23;
   }
}
...
```

Estruturas de apoio

Aplicações escritas em linguagem C podem usufruir de estruturas definidas no arquivo de cabeçalho *ScopeApi.h*. O buffer recebido contendo a lista de operadoras resume-se na estrutura abaixo:

```
typedef struct
{
   short NumOperCel;
   char  OperCel[2000];
} stREC_CEL_OPERADORAS, *ptREC_CEL_OPERADORAS;
```

, onde:

- *NumOperCel*: número de operadoras de celular retornadas nesta transação;
- *OperCel*: tabela de operadoras.

No entanto, o membro `OperCel` da estrutura acima contém a lista desejada e para percorrê-la pode-se utilizar a estrutura:

```
typedef struct
{
   char CodOperCel;
   char NomeOperCel[TAM_NOME_OP];
```

```
} stREC_CEL_ID_OPERADORA, *ptREC_CEL_ID_OPERADORA;
```

, onde:

- *CodOperCel*: código, da operadora, que será escolhido pelo operador e devolvido para o SCOPE;
- *NomeOperCel*: nome da operadora representada por uma string.

Reescrevendo o código exemplificado acima, obtemos:

```
int i;
stREC_CEL_OPERADORAS ListaOper;
stREC_CEL_ID_OPERADORA Oper;
long retorno;
...
retorno = ScopeRecuperaOperadorasRecCel(2,
                                  (char *) &ListaOper, sizeof(ListaOper));
if(retorno == 0)
{
   printf("\n\n LISTA DE OPERADORAS:\n\n");
   /* Exibe as operadoras */
   for(i = 0; I < (int)ListaOper.NumOperCel; i++)
   {
      // inicializa estruturas
      memcpy(&Oper, &ListaOper.OperCel[i*sizeof(stREC_CEL_ID_OPERADORA)],
                                  sizeof(stREC_CEL_ID_OPERADORA));
      printf("\nCod %d  - Oper. %.21s",
                                  (int) buffer[ind_op], buffer[ind_op+1]);
   }
}
```

Protótipo

```
LONG EXPORT ScopeRecuperaOperadorasRecCel (BYTE TpTab, char *buffer,
WORD TamBuf)
```

Parâmetros

[in]	BYTE	TpTab	Informa o formato da tabela que a aplicação deseja receber a lista de operadoras. Aceita apenas o valor 2.
[out]	String	Buffer	Buffer com a lista de operadoras.
[in]	WORD	TamBuf	Tamanho do buffer anterior, previamente alocado, que deve ser maior ou igual a 2002 bytes.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido

0xFA02	64002	Parâmetro 2 inválido
0xFA03	64003	Parâmetro 3 inválido

Exemplo

```
...
stREC_CEL_OPERADORAS ListaOper;
...
// obtém status do SCOPE
switch(status_scope)
{
   ...
   case 64624:
      ScopeRecuperaOperadorasRecCel(2, (char *)&ListaOper, sizeof(buffer));
      ApresentaListaAoOperador(ListaOper);
      // Obtém código da operadora escolhido pelo cliente
      break;
   ...
}
// passa a informação coletada para o SCOPE
...
```

Obtendo valores de recarga

Com o status do SCOPE igual a 64558, a aplicação deverá informar as opções de recarga disponíveis para a operadora escolhida, que são disponibilizadas pela função ScopeRecuperaValoresRecCel() de maneira similar à função que recupera as operadoras. A estrutura do buffer recebido é a seguinte:

<table>
<tr><td>1 byte</td><td colspan="2">Um caractere representando o tipo de valor.

<table><tr><th>Valor (ASCII)</th><th>Significado</th></tr><tr><td>‘V’</td><td>Valor variável</td></tr><tr><td>‘F’</td><td>Valor Fixo</td></tr><tr><td>‘T’</td><td>Todos valores</td></tr></table></td></tr>
<tr><td>12 bytes</td><td colspan="2">Valor mínimo representado por string</td></tr>
<tr><td>12 bytes</td><td colspan="2">Valor máximo representado por string</td></tr>
<tr><td rowspan="5">1 byte</td><td colspan="2">Total de valores fixos da tabela</td></tr>
<tr><td>12 bytes</td><td>Valor fixo 1 da recarga, com 2 casas decimais</td></tr>
<tr><td>12 bytes</td><td>Bônus da recarga para o valor 1</td></tr>
<tr><td>12 bytes</td><td>Custo da recarga para o valor 1</td></tr>
<tr><td>12 bytes</td><td>Valor fixo 2 da recarga, com 2 casas decimais</td></tr>
</table>

	12 bytes	Bônus da recarga para o valor 2
	12 bytes	Custo da recarga para o valor 2
	12 bytes	Valor fixo 3 da recarga, com 2 casas decimais
	12 bytes	Bônus da recarga para o valor 3
	12 bytes	Custo da recarga para o valor 3
	12 bytes	Valor fixo 4 da recarga, com 2 casas decimais
	12 bytes	Bônus da recarga para o valor 4
	12 bytes	Custo da recarga para o valor 4
	12 bytes	Valor fixo 5 da recarga, com 2 casas decimais
	12 bytes	Bônus da recarga para o valor 5
	12 bytes	Custo da recarga para o valor 5
	12 bytes	Valor fixo 6 da recarga, com 2 casas decimais
	12 bytes	Bônus da recarga para o valor 6
	12 bytes	Custo da recarga para o valor 6
	12 bytes	Valor fixo 7 da recarga, com 2 casas decimais
	12 bytes	Bônus da recarga para o valor 7
	12 bytes	Custo da recarga para o valor 7
	12 bytes	Valor fixo 8 da recarga, com 2 casas decimais
	12 bytes	Bônus da recarga para o valor 8
	12 bytes	Custo da recarga para o valor 8
	12 bytes	Valor fixo 9 da recarga, com 2 casas decimais
	12 bytes	Bônus da recarga para o valor 9
	12 bytes	Custo da recarga para o valor 9
	12 bytes	Valor fixo 10 da recarga, com 2 casas decimais
	12 bytes	Bônus da recarga para o valor 10
	12 bytes	Custo da recarga para o valor 10
41 bytes	Mensagem promocional a ser exibida com os valores	

Estruturas de apoio

Definidas no arquivo de cabeçalho *ScopeApi.h*, estão as estruturas abaixo, que são utilizadas para o recebimento do buffer de valores. A primeira delas:

```
typedef struct
{
   char TipoValor;
   char ValorMinimo[12];
   char ValorMaximo[12];
   char Totvalor;
   stREC_CEL_VALOR TabValores[10];
   char MsgPromocional[41];
} stREC_CEL_VALORES, *ptREC_CEL_VALORES;
```

, onde:

- TipoValor: caractere que representa o tipo de valor permitido para a operadora e que define quais valores serão exibidos. O domínio de valores para este campo é:
    - ‘V’ – valor variável entre o que está indicado no campo ValorMinimo e ValorMaximo desta mesma estrutura
    - ‘F’ – valores fixos que estão no campo TabValores
    - ‘T’ – disponível tanto o valor variável quanto o fixo
- ValorMinimo: este campo é uma cadeia com 12 caracteres numéricos que representam o valor mínimo do intervalo de valores variáveis com a vírgula implícita (exemplo: caso o dado neste campo seja 000000001000, então ele está representando o valor R$10,00);
- ValorMaximo: este campo é uma cadeia com 12 caracteres numéricos que representam o valor máximo do intervalo de valores variáveis com a vírgula implícita (exemplo: caso o dado neste campo seja 000000001500, então ele está representando o valor R$15,00);
- Totvalor: este campo é binário e indica a quantidade de valores fixos que estão no campo TabValores;
- TabValores: cadeia de registros, descrita abaixo, com todos os valores fixos disponíveis para a recarga desta transação que, embora seja fixo com 10 registros, deve-se sempre observar a quantidade válida representada pelo campo anterior;
- MsgPromocional: string contendo uma mensagem promocional a ser exibida com os valores de recarga.

CUIDADO: como a estrutura de valores é sempre fixa com dez registros de valores e por esses vários valores do domínio do campo de tipo de valor, é extremamente recomendável que a aplicação sempre verifique a quantidade de valores fixos, informada pelo campo TotValor, disponíveis.

Como se pode notar pela definição, o campo TabValores é do tipo da estrutura stREC_CEL_VALOR, descrita abaixo.

```
typedef struct
{
   char Valor[12];
   char Bonus[12];
   char Custo[12];
} stREC_CEL_VALOR, *ptREC_CEL_VALOR;
```

- Valor: este campo é uma cadeia com 12 caracteres numéricos que representam um valor fixo, para escolha do cliente, com a vírgula implícita (exemplo: caso o dado neste campo seja 000000001500, então ele está representando o valor R$15,00);

- Bônus: este campo é uma cadeia com 12 caracteres numéricos que representam o bônus que o cliente ganha ao escolher o valor fixo deste registro com a vírgula implícita (exemplo: caso o dado neste campo seja 000000010000, então ele está representando o valor R$100,00);
- Custo: este campo é uma cadeia com 12 caracteres numéricos que representam o custo da recarga ao escolher o valor fixo deste registro com a vírgula implícita (exemplo: caso o dado neste campo seja 000000005000, então ele está representando o valor R$50,00);

```
typedef struct
{
   char TipoValor;
   char ValorMinimo[10];
   char ValorMaximo[10];
   char Totvalor;
   stREC_CEL_VALOR TabValores[10];
   char MsgPromocional[41];

   char TotFaixaValores;
      stREC_CEL_FAIXA_VALORES_2 TabFaixaValores[10];
} stREC_CEL_VALORES_MODELO_4, *ptREC_CEL_VALORES_MODELO_4;
```

A rede GWCel permite que o cliente faça uma recarga, utilizando uma faixa de valores, ao invés de valores pré-definidos. A maioria dos membros dessa estrutura são os mesmos da stREC_CEL_VALORES, sendo adicionais:

- TotFaixaValores: indica a quantidade total de faixas de valores disponíveis;
- TabFaixaValores: cadeia de registros, descrita abaixo, com todos as faixas de valores disponíveis para a recarga desta transação que, embora seja fixo com 10 registros, deve-se sempre observar a quantidade válida representada pelo campo anterior;

Existe uma diferença na estrutura dos dados enviados pela GWCel dependendo a versão de sua especificação.

Para a rede GWCel versão 003 a estrutura de Valores de Faixa é a seguinte:

```
typedef struct
{
    char ValorMin[10];
    char ValorMax[10];
} stREC_CEL_FAIXA_VALORES, *ptREC_CEL_FAIXA_VALORES;
```

- ValorMin: indica o valor mínimo aceitável para a recarga;
- ValorMax: indica o valor máximo aceitável para a recarga.

Para a Gwcel versão 005, as estruturas de Valores de Faixa também possuem os campos Bonus e Custo, antes só presentes na estrutura de valores fixos:

```
typedef struct
{
    char ValorMin[12];
    char ValorMax[12];
    char Bonus[12];
    char Custo[12];

} stREC_CEL_FAIXA_VALORES_2, *ptREC_CEL_FAIXA_VALORES_2;
```

- ValorMin: indica o valor mínimo aceitável para a recarga;
- ValorMax: indica o valor máximo aceitável para a recarga.
- Bonus: Indica o valor do bônus que o cliente receberá ao escolher a faixa.
- Custo: Indica o custo da recarga.

Protótipo

```
LONG EXPORT ScopeRecuperaValoresRecCel (BYTE TpTab, char *buffer, WORD TamBuf)
```

Parâmetros

[in]	BYTE	TpTab	Informa o formato da tabela que a aplicação deseja receber a lista de operadoras. Aceita apenas o valor 2.
[out]	String	Buffer	Buffer com a lista de valores de recarga disponíveis.
[in]	WORD	TamBuf	Tamanho do array buffer (segundo parâmetro) alocado que deve ser igual a 427 bytes.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFA03	64003	Parâmetro 3 inválido

Exemplo

```
...
stREC_CEL_VALORES tabVal;
...
// obtém status do SCOPE
switch(status_scope)
{
   ...
   case 64558:
      ScopeRecuperaValoresRecCel(2, (char *)&tabVal, sizeof(buffer));
```

```
      ApresentaValoresAoOperador(tabVal);
      // Obtém o valor de recarga escolhido pelo cliente
      break;
   ...
}
// passa a informação coletada para o SCOPE
...
```

CUIDADO: os campos de valores representados nas estruturas de recarga não são *strings* terminadas com o caractere *nulo*, mas são cadeias de caracteres de tamanho fixo igual a 12 com zeros à esquerda.

Funções de consulta para recarga de celular

O Scope Client não disponibilizava funções de consulta a planos de recarga de celular, úteis para sistemas de retaguarda que desejavam vender pacotes de recarga antecipadamente dentro de um pré-pedido. Para contornar isso, foram adicionadas duas funções, uma para retornar as operadoras e outra para retornar os valores disponíveis para recarga de uma operadora de uma localidade específica.

Obtendo operadoras disponíveis fora do fluxo de recarga

Para obter as operadoras disponíveis, deve ser feita uma chamada a função ScopeObtemOperadorasRecCelOffTEF(),ela retornará um buffer com as operadoras disponíveis .

Protótipo

```
LONG EXPORT ScopeObtemOperadorasRecCelOffTEF (BYTE _TipoTabela, char *_Buffer, WORD _TamBuffer)
```

Parâmetros

[in]	BYTE	_TipoTabela	Informa o formato da tabela que a aplicação deseja receber a lista de operadoras. Aceita apenas o valor 2.
[out]	String	_Buffer	Buffer com a lista de operadoras.
[in]	WORD	_TamBuffer	Tamanho do buffer anterior, previamente alocado, que deve ser maior ou igual a 2002 bytes.

Retorno

Ver tabela de código de retorno.

Exemplo

```
...
LONG RC;
stREC_CEL_OPERADORAS       ListaOper     = {0};
stREC_CEL_ID_OPERADORA     OPER;
...
// Obtem as operadoras.
```

```
 RC = ScopeObtemOperadorasRecCelOffTEF(REC_CEL_OPERADORAS_MODELO_2,(P_CHAR)
&ListaOper, sizeof (ListaOper);
if(RC =! != RCS_SUCESSO)
{
    return RC;
}
   ...
// exibe as operadoras
```

Obtendo valores de recarga disponíveis fora do fluxo de recarga

Para obter os valores disponíveis para recarga, deve ser feita uma chamada a função ScopeRecuperaValoresRecCelOffTEF() ela retornará um buffer com a lista de valores disponíveis para uma operadora de uma localidade específica.

Protótipo

```
LONG EXPORT ScopeRecuperaValoresRecCelOffTEF(BYTE _TipoTabela, char
*_Buffer, WORD _TamBuffer, char* _CodOperadora, char* _CodLocalidade)
```

Parâmetros

[in]	BYTE	_TipoTabela	Informa o formato da tabela que a aplicação deseja receber a lista de operadoras. Aceita apenas o valor 2.
[out]	String	_Buffer	Buffer com a lista de valores de recarga disponíveis.
[in]	WORD	_TamBuffer	Tamanho do array buffer (segundo parâmetro) alocado que deve ser igual a 427 bytes.
[in]	String	_CodOperadora	Código da bandeira da operadora
[in	String	_CodLocalidade	Código da localidade (DDD)

Retorno

Ver tabela de código de retorno.

Exemplo

```
...
LONG RC;
char  codOperadora[3 + 1];
char  codLocalidade[2 + 1];
stREC_CEL_VALORES ListaValores = {0};
...
// Entra com o código da operadora e a localidade

RC=ScopeRecuperaValoresRecCelOffTEF((BYTE)REC_CEL_VALORES_MODELO_2,(P_CHAR)
&ListaValores, sizeof (ListaValores),codOperadora,codLocalidade );
if ( RC == RCS_SUCESSO || RC == RCS_ACQUIRER_OFF)
{
// Exibe os valores
}
```

...

Estorno de transações

O estorno é uma transação de anulação de outra transação que já está confirmada no lado da autorizadora e pode ser realizada quando ocorrerem erros na digitação (valor, data de agendamento das parcelas, numero de parcelas, etc.) ou desistência da compra por parte do cliente. O SCOPE disponibiliza uma única função para o estorno das diversas transações: ScopeCancelamento(). No entanto, nem todas as transações são estornáveis, como por exemplo, transações de consultas. Na tentativa de estorno destas transações, o SCOPE retornará através da função ScopeStatus() o código de erro 65286, informando que a transação não é cancelável.
O estorno difere do desfazimento no sentido em que o primeiro é uma nova transação isolada, enquanto o segundo é a finalização de uma transação. Tanto, que existe o desfazimento do estorno, que não permite que ocorra o cancelamento da transação original, mantendo-a com o status de confirmada.

CUIDADO: não é possível o cancelamento de uma transação sem que esta já tenha sido confirmada. Em outras palavras, no SCOPE não se pode cancelar uma transação que ainda está numa sessão de TEF em aberto. Para este cancelamento a transação deve ser desfeita (ver Encerrando a sessão).

Estornando a transação

Para a realização do estorno, é necessário informar ao SCOPE os dados da transação original, e, em muitas vezes, o número do controle gerado pelo próprio SCOPE na transação.

Protótipo

```
LONG EXPORT ScopeCancelamento (char *Valor, char *TxServico)
```

Parâmetros

[in]	String com o máximo de 12 dígitos	Valor	Valor da transação com a vírgula implícita (exemplo: R$ 123,00 = “12300”)
[in]	String	TxServico	Valor da taxa de serviço

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado

0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
char valor[12 + 1];
char taxa[12 + 1];
...
// obtém o valor da compra e armazena em 'valor'
// obtém a taxa de serviço e armazena em 'taxa'
// abre sessão
...
retorno = ScopeCancelamento(valor, taxa);
...
// processa o cancelamento
...
// fecha a sessão
...
```

Comprovantes de transações

Qualquer transação realizada com sucesso e com algum tipo de comprovante para a impressão deve ser obtido pela função ScopeGetCupomEx().

IMPORTANTE: quando a aplicação utiliza a interface HLAPI, ela sempre deve chamar a função ScopeSuspend() passando como parâmetro o valor inteiro 4 (ver Suspendendo a transação).

Obtendo os cupons de TEF

Durante o processamento de uma transação, a aplicação deverá esperar pelos códigos de coleta:

- 64582 (0xFC46): imprimir cupom parcial;
- 64514 (0xFC02): imprimir cupom;
- 64539 (0xFC1B): imprimir consulta;
- 64546 (0xFC22): imprime nota promissória;
- 64593 (0xFC51): imprime cupom promocional.

Neste momento, a aplicação chamará a função para obter os cupons da transação.

CUIDADO: quando a aplicação realizar uma consulta CDC no meio de uma TEF de débito, a aplicação não deverá enviar o cupom da consulta para a impressora, mas apenas exibi-lo na tela. No entanto, quando for apenas uma transação de consulta, isto é, foi chamada apenas uma função do tipo ScopeConsulta<*transação*>(), o comprovante recebido deve ser enviado para a impressora.

Protótipo

```
LONG EXPORT ScopeGetCupomEx(WORD CabecLen, char *Cabec,
                            WORD CupomClienteLen, char *CupomCliente,
                            WORD CupomLojaLen, char *CupomLoja,
                            WORD CupomReduzLen, char *CupomReduz,
                            BYTE *NroLinhasReduz)
```

Parâmetros

[in]	WORD	CabecLen	Tamanho reservado pela aplicação de PDV para receber o cabeçalho do cupom
[out]	String	Cabec	Ponteiro para área onde será recebido o cabeçalho do cupom
[in]	WORD	CupomClienteLen	Tamanho reservado pela aplicação de PDV para receber a via do cupom Cliente
[out]	String	CupomCliente	Ponteiro para área onde será recebida a via do cliente que sempre estará

			disponível para a aplicação de PDV
[in]	WORD	CupomLojaLen	Tamanho reservado pela aplicação para receber a via da loja
[out]	String	CupomLoja	Ponteiro para área onde será recebida a via da loja que sempre estará disponível para a aplicação
[in]	WORD	CupomReduzLen	Tamanho reservado pelo aplicativo para receber o cupom reduzido
[out]	String	CupomReduz	Ponteiro para área onde será recebida a via cupom reduzido que poderá substituir a via do cliente e que em alguns casos, esta via pode não estar disponível para a aplicação
[out]	WORD	NroLinhasReduz	Número de linhas que tem o cupom reduzido. Se o valor for zero, não há cupom reduzido disponível

IMPORTANTE: Os finalizadores de linha dos cupons retornados pela função ScopeGetCupomEx estão de acordo com a parametrização do ScopeForneceCampo(SCOPE_DADO_SEPARADOR_LINHA).

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE04	65028	Não existe transação suspensa
0xFF12	65298	Área reservada para o buffer é insuficiente para o SCOPE Client x os dados solicitados

Exemplo

```
...
BYTE NroLnhReduzido = 0;
char Cabec[1024],
     CpCliente[2048],
     CpLoja[2048],
     CpReduzido[2048];
LONG retorno;
...
// obtém status do SCOPE
switch(status_scope)
{
   ...
   case 64582:
   case 64514:
   case 64539:
   case 64546:
   case 64593:
```

```
      memset(Cabec, 0, sizeof(Cabec));
      memset(CpCliente, 0, sizeof(CpCliente));
      memset(CpLoja, 0, sizeof(CpLoja));
      memset(CpReduzido, 0, sizeof(CpReduzido));
      retorno = ScopeGetCupomEx(sizeof(Cabec), Cabec,
                                sizeof(CpCliente), CpCliente,
                                sizeof(CpLoja), CpLoja,
                                sizeof(CpReduzido), CpReduzido,
                                &NroLnhReduzido);
      if (RC == RCS_SUCESSO)
      {
         ImprimeCupomTEF("CABECALHO", Cabec);
         ImprimeCupomTEF("CUPOM DO CLIENTE", CpCliente);
         ImprimeCupomTEF("CUPOM DA LOJA", CpLoja);
         if (NroLnhReduzido > 0)
            ImprimeCupomTEF("CUPOM REDUZIDO", CpReduzido);
      }
      break;
   ...
}
...
```

Reimpressão de comprovante

Conforme necessidade, o operador pode solicitar uma nova cópia do comprovante da última transação realizada ou alguma específica anterior a última, desde que tenha sido aprovada e se encontre no SCOPE. São duas as modalidades de reimpressão: *on-line* e *off-line*. Essa necessidade pode ter sido ocasionada por algum problema com a impressão original (cupom ilegível, papel enroscado na impressora, etc.).

As duas modalidades de reimpressão do comprovante é uma transação e como tal, deverá ser tratada numa sessão de TEF, preferencialmente, sendo a única da sessão.

Solicitando o comprovante off-line

Para toda e qualquer rede, o SCOPE permite reimprimir apenas o último comprovante realizado num PDV. É possível reimprimir cupons antigos, exceto o reduzido, desde que estes estejam na base de dados do SCOPE. Esta reimpressão atua apenas entre o SCOPE Client e o SCOPE Server e não vai para a autorizadora.

Protótipo

```
LONG EXPORT ScopeReimpressaoOffLine (void)
```

Parâmetros

Não há parâmetros.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
*0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
// abre sessão
...
retorno = ScopeReimpressaoOffLine ();
...
// processa a transação
...
// fecha a sessão
...
```

IMPORTANTE: na reimpressão não há cupom reduzido.

IMPORTANTE: a função ScopeReimpressaoOffLine() substitui a anterior ScopeReimpressaoComprovante(). No entanto, esta última ainda é mantida por compatibilidade.

Solicitando o comprovante on-line

Utilizada nos casos em que a rede autorizadora ou o correspondente bancário oferece a transação (on-line) de reimpressão do comprovante do pagamento de conta.

Protótipo

```
LONG EXPORT ScopeReimpressaoOnLine (WORD CodBandeira)
```

Parâmetros

[in]	WORD	CodBandeira	Código da bandeira do cartão (ver Código das bandeiras)

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
*0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
WORD CodBandeira;
// abre sessão
...
// Obtém a bandeira do cartão
...
retorno = ScopeReimpressaoOnLine(CodBandeira);
...
// processa a transação
...
// fecha a sessão
...
```

IMPORTANTE: para a certificação da Visanet 4.1 esta função não é necessária estar implementada.

IMPORTANTE: a função ScopeReimpressaoOnLine() substitui a anterior ScopeReimpressaoComprovantePagamento(). No entanto, esta última ainda é mantida por

compatibilidade.

Imprimindo o comprovante correto

Como acontece em qualquer outra transação, na reimpressão de comprovante, os cupons são recebidos pela função de cupons (ver Comprovantes de transações), a qual retorna todos os cupons. Entretanto, devido exigência da Visanet, a aplicação deverá imprimir apenas o cupom solicitado pelo operador. Para isso, a aplicação deverá consultar no SCOPE qual é o cupom que deverá ser impresso, com a ajuda da função ScopeObtemTipoViaReimpressao().

A necessidade desta função surgiu por causa da interface HLAPI, porque o SCOPE coletava do operador qual a via que deveria ser impressa, mas a aplicação, que envia os comprovantes para a impressora, não sabia dessa escolha.

Protótipo

```
LONG EXPORT ScopeObtemTipoViaReimpressao (BYTE *EhReimpre, BYTE *Via)
```

Parâmetros

[out]	BYTE	EhReimpre	Informa se é uma transação de reimpressão (valor = 1) ou qualquer outra transação (valor = 0)
[out]	BYTE	Via	Caso seja uma transação de reimpressão, informa qual a via a ser impressa: - 0: todas as vias - 1: apenas a via da loja - 2: apenas a via do cliente

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE04	65028	Não existe transação suspensa
0xFF11	65297	Não existe cupom válido

Exemplo

```
...
BYTE NroLnhReduzido = 0,
     EhReimpressao,
     ViaReimpressao;
char Cabec[1024],
     CpCliente[2048],
     CpLoja[2048],
     CpReduzido[2048];
LONG retorno;
...
// obtém status do SCOPE
switch(status_scope)
{
```

```
...
case 64582:
case 64514:
case 64539:
case 64546:
case 64593:
   memset(Cabec, 0, sizeof(Cabec));
   memset(CpCliente, 0, sizeof(CpCliente));
   memset(CpLoja, 0, sizeof(CpLoja));
   memset(CpReduzido, 0, sizeof(CpReduzido));
   retorno = ScopeGetCupomEx(sizeof(Cabec), Cabec,
                             sizeof(CpCliente), CpCliente,
                             sizeof(CpLoja), CpLoja,
                             sizeof(CpReduzido), CpReduzido,
                             &NroLnhReduzido);
   if (retorno == RCS_SUCESSO)
   {
      retorno = ScopeObtemTipoViaReimpressao( &EhReimpressao,
                                              &ViaReimpressao );
      if(EhReimpressao == 0)
      {
         ImprimeCupomTEF("CABECALHO", Cabec);
         ImprimeCupomTEF("CUPOM DO CLIENTE", CpCliente);
         ImprimeCupomTEF("CUPOM DA LOJA", CpLoja);
         if (NroLnhReduzido > 0)
            ImprimeCupomTEF("CUPOM REDUZIDO", CpReduzido);
      }
      else
      {
         switch(ViaReimpressao)
         {
         case 0:
            ImprimeCupomTEF("CUPOM DO CLIENTE", CpCliente);
            ImprimeCupomTEF("CUPOM DA LOJA", CpLoja);
            break;
         case 1:
            ImprimeCupomTEF("CUPOM DO CLIENTE", CpCliente);
            break;
         case 2:
            ImprimeCupomTEF("CUPOM DA LOJA", CpLoja);
            break;
         }
      }
   }
   break;
```

```
    ...
}
...
```

PBM - Medicamentos

Aqui trataremos do grupo de funções que atendem a funcionalidade de PBM.

Consultando medicamento

A função ScopeConsultaMedicamento() aciona uma transação no SCOPE para obter uma lista de medicamentos relacionados a uma autorização, assim como a quantidade autorizada, o PMC (Preço Máximo para o Consumidor) e o preço e-Pharma para cada um destes.
O processamento segue a linha de uma transação qualquer de TEF, em que a aplicação deve processar através das chamadas à função de consulta status (ver Status de transação) e coleta de dados. O PDV deve esperar pelo estado 64580 para chamar a função específica e obter a lista de medicamentos (ver Lista de medicamentos, Lista de medicamentos com CRM e Lista de medicamentos Ex).

Protótipo

```
LONG EXPORT ScopeConsultaMedicamento(BYTE TipoConvenio, BYTE CodigoRede)
```

Parâmetros

[in]	BYTE	TipoConvenio	Tipo do convênio. (0=PBM, 1=Empresa)
[in]	BYTE	CodigoRede	Código da rede (ver Convênios)

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
BYTE    TipoConvenio, CodigoRede;
...
// abre sessão
...
// coleta o tipo do convênio, o código da rede
...
retorno = ScopeConsultaMedicamento (TipoConvenio, CodigoRede);
...
// processa a transação
...
// fecha a sessão
...
```

Nesse tipo de transação, o SCOPE não possui controle do valor de venda a ser registrado. Portanto, ao receber o código 64579 da função de status, a aplicação deve atualizar o valor (ver ScopeAtualizaValor), informando assim o valor da venda.

Exemplo

```
char VlOperac [12+1];
...
switch(status_scope)
{
   ...
   Case 64579:
        RecebeValorDoOperador (&VlOperac);
        iRet = ScopeAtualizaValor (VlOperac);
        if (iRet != RCS_SUCESSO)
            // Tratar erro
        break;
   ...
```

Compra de medicamento

Para iniciar uma compra de medicamentos PBM, deve-se chamar a função ScopeCompraMedicamento(). O comportamento se assemelha à consulta de medicamento e também pode retornar no fluxo de coleta o código de coleta 64580 para obter a lista de medicamentos.

Protótipo

```
LONG EXPORT ScopeCompraMedicamento(BYTE TipoConvenio,
                                   BYTE CodigoRede,
                                   char *NumCpFiscal)
```

Parâmetros

[in]	BYTE	TipoConvenio	Tipo do convênio. (0=PBM, 1=Empresa)
[in]	BYTE	CodigoRede	Código da rede (ver Convênios)
[out]	String	NumCpFiscal	Número do cupom fiscal para o PDV

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	**Decimal**	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFA03	64003	Parâmetro 3 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
BYTE    TipoConvenio, CodigoRede;
char   *NumCpFiscal;
...
// abre sessão
...
// coleta o tipo do convênio, o código da rede e o número do cupom fiscal
...
retorno = ScopeCompraMedicamento (TipoConvenio, CodigoRede, NumCpFiscal);
...
// processa a transação
...
// fecha a sessão
...
```

Funcionamento a partir do layout 5

A mudança para o layout 5 (conforme descrito no documento “Scope PBM - Versão 1.10.doc”) trouxe mudanças significativas no funcionamento da compra de medicamentos. Para realizar uma compra de medicamentos, é preciso obrigatoriamente ter feito uma consulta de medicamentos antes e estar na mesma sessão TEF.

O motivo para tal funcionamento é que a partir do layout 5, pode ser requisitado coleta de senha através do bit 22 do retorno da consulta (mensagem 0110). Como essa informação referente à necessidade de coleta de senha deve ser coletada em uma transação diferente (0200), ela é armazenada em um arquivo de contexto e recuperado na compra de medicamentos. Atualmente, as redes Vidalink e PBM Padrão trabalham com o layout na nova versão.

Lista de medicamentos

Num fluxo de compra ou consulta de medicamento, recebendo o código de coleta 64580 (ver Status de transação), a aplicação deverá obter a lista de medicamentos disponíveis, através das funções ScopeObtemMedicamentos(), ScopeObtemMedicamentosComCRM() ou ScopeObtemMedicamentosEx(), que serão tratadas logo abaixo. O formato da lista obtido pela função ScopeObtemMedicamentos() encontra se na tabela abaixo.

Posição	Formato	Descrição
01 a 13	String	Código EAN do medicamento.
14 a 15	String	Quantidade autorizada do produto.
16 a 22	String	Preço máximo ao consumidor.
23 a 29	String	Preço de venda.
30 a 36	String	Preço de fábrica.
37 a 43	String	Preço de aquisição.
44 a 50	String	Preço de repasse.
51	Byte	Reservado para uso futuro.
52 a 53	String	Motivo da rejeição.

Protótipo

```
LONG EXPORT ScopeObtemMedicamentos(BYTE *Qtd,
                                   char *ListaMedicamentos,
                                   WORD TamLista)
```

Parâmetros

[out]	BYTE	Qtd	Retorna a quantidade de medicamentos consultados ou comprados
[out]	String	ListaMedicamentos	Retorna os medicamentos consultados ou comprados
[in]	WORD	TamLista	Tamanho, em bytes, do campo

			“ListaMedicamentos”

Retorno
Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE04	65028	Não existe transação suspensa
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV’s disponíveis. Erro retornado quando a aplicação passa o caractere “#” como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV’s disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
BYTE bQtdVias = 0;
LONG lRet;
stREGISTRO_MEDICAMENTO lstMedsCRM [38] = {0};

lRet = ScopeObtemMedicamentos
            ( &bQtdVias, (char *) &lstMedsCRM, sizeof (lstMedsCRM) );

if (lRet == RCS_SUCESSO)
{
    /* Sucesso no recebimento dos medicamentos,
       os membros das estruturas estarão preenchidos */
}
else
    // Erro ...
```

Lista de medicamentos com CRM

A função ScopeObtemMedicamentosComCRM() recupera a lista dos medicamentos consultados ou comprados com o CRM do médico. Esta função não suporta as bandeiras Novartis e FlexMed e seu formato obedece à definição da tabela abaixo.

Posição	Formato	Descrição
01 a 53	String	Dados do medicamento, obedecendo a definição da tabela retornada para a função ScopeObtemMedicamentos().
54 a 62	String	Número do CRM.

Protótipo

```
LONG EXPORT ScopeObtemMedicamentosComCRM(BYTE *Qtd,
                                        BYTE *TipoConv,
                                        char *ListaMedicamentos,
                                        WORD TamLista)
```

Parâmetros

[out]	BYTE	Qtd	Retorna a quantidade de medicamentos consultados ou comprados
[out]	BYTE	TipoConv	Retorna o tipo de convênio: 0: Pagamento à vista; 1: Pagamento a prazo; 2: Empresa Fechada; 3: Empresa Aberta.
[out]	String	ListaMedicamentos	Retorna os medicamentos consultados ou comprados
[in]	WORD	TamLista	Tamanho, em bytes, do campo “ListaMedicamentos”

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE04	65028	Não existe transação suspensa
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV’s disponíveis. Erro retornado quando a aplicação passa o caractere “#” como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV’s disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF11	65297	Não existe cupom válido
0xFF12	65298	Área reservada para o buffer é insuficiente para o SCOPE Client x os dados solicitados
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Lista de medicamentos Extendida

A função ScopeObtemMedicamentosEx() recupera a lista dos medicamentos consultados ou comprados em qualquer formato disponível atualmente. A função deve receber qual a versão que se deseja da lista de medicamentos como será descrito a seguir. Os formatos de listas antigas permanecem os mesmos, porém, há o formato do layout 5 (conforme descrito no documento “Scope PBM - Versão 1.10.doc”) que obedece à tabela abaixo:

Posição	Formato	Descrição
01 a 13	String	Código EAN do medicamento.
14 a 15	String	Quantidade autorizada do produto.
16 a 17	String	Motivo da rejeição.
18 a 26	String	Código do convênio.
27 a 34	String	Preço máximo ao consumidor.
35 a 42	String	Preço de venda.
43 a 50	String	Preço de fábrica.
51 a 58	String	Preço de aquisição.
59 a 66	String	Preço de repasse.
67 a 79	String	Número do CRM.

Protótipo

```
LONG EXPORT ScopeObtemMedicamentosEx(BYTE IdLayout,
                                     BYTE *_QtdRegistros,
                                     char *_ListaMedicamentos,
                                     WORD _TamLista,
                                     BYTE *_TipoConvenio)
```

Parâmetros

[in]	BYTE	IdLayout	Versão da lista de medicamentos que se deseja recuperar, devendo ser como definido na tabela a seguir.
[out]	BYTE	QtdRegistros	Retorna a quantidade de medicamentos consultados ou comprados.
[out]	String	ListaMedicamentos	Retorna os medicamentos consultados ou comprados.
[in]	WORD	TamLista	Tamanho, em bytes, do campo “ListaMedicamentos”
[out]	BYTE	TipoConvenio	Retorna o tipo de convênio: 0: Pagamento à vista; 1: Pagamento a prazo; 2: Empresa Fechada; 3: Empresa Aberta.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	**Decimal**	
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE04	65028	Não existe transação suspensa
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Layouts:

ID layout	Usar estrutura
1	stREGISTRO_MEDICAMENTO
2	stREGISTRO_MEDICAMENTO_CRM
3	stREGISTRO_MEDICAMENTO_L3
4	stREGISTRO_PHARMASYSTEM_RET

IMPORTANTE: Atualmente, somente a rede Vidalink e PBM Padrão é compatível com o layout 3 de estrutura.

Lista de Projetos

A função ScopeRecuperaBufTabela() fornecerá para a aplicação de automação comercial uma lista de informações a serem exibidas na tela para facilitar a seleção da opção desejada. Ela servirá para facilitar futuras implementações que atendam a mesma finalidade. O parâmetro _TipoTabela indicará o formato dos dados que serão fornecidos.

```
LONG EXPORT ScopeRecuperaBufTabela  (BYTE TipoTabela,
                                     char *_QtdRegistros,
                                     char *_Buffer,
                                     WORD _TamBuffer)
```


Parâmetros

[in]	BYTE	TipoTabela	Tipo de lista de elementos que se deseja recuperar, devendo ser como definido na tabela a seguir.
[out]	String	QtdRegistros	Retorna a quantidade de elementos da tabela
[out]	String	Buffer	Retorna os elementos da tabela.
[in]	WORD	TamBuffer	Tamanho, em bytes, do campo “Buffer”

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFA03	64003	Parâmetro 3 inválido

Layouts:

ID layout	Usar estrutura	Descrição
BUF_TAB_PRJ_PHARMASYSTEM (ID 0)	stREGISTRO_PROJETO_PHMS	Projeto PharmaSystem

O formato obedece à tabela, conforme a especificação da PharmaSystem

Posição	Formato	Descrição
01 a 06	String	Código do Projeto
07 a 36	String	Descrição do Projeto
37 a 56	String	Operadora do Projeto

Elegibilidade do Cartão PBM

Para iniciar uma compra de medicamentos PDM, sem a existência de uma autorização previamente fornecida por uma aplicação de balcão de farmácia, deve-se chamar a função ScopeElegibilidadeCartao(), que fará a validação dos dados do cliente que deseja efetuar a compra. Essa função suporta a bandeira PharmaSystem.

Protótipo

```
LONG EXPORT ScopeElegibilidadeCartao (BYTE _TipoConvenio,
```

```
                                        BYTE _CodRede,
                                        char *NumCpFiscal)
```

Parâmetros

[in]	BYTE	_TipoConvenio	Tipo do convênio. (0=PBM, 1=Empresa)
[in]	BYTE	_CodRede	Código da rede (ver Convênios)

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
BYTE    _TipoConvenio, _CodigoRede;
...
// abre sessão
...
// coleta o tipo do convênio, o código da rede
...
retorno = ScopeElegibilidadeCartao (_TipoConvenio, _CodigoRede);
...
// processa a transação
...
// fecha a sessão
...
```

Pre-Autorizacao de Medicamentos PBM

Para informar cada produto que será incluído na compra, obtendo informações do mesmo, deve ser chama a função ScopePreAutorizacaoMedicamento(). A função pode retornar no fluxo de coleta o código de coleta 64579 para fornecer a lista de medicamentos, e o código de coleta 64580 para obter a lista de medicamentos. Essa função suporta a bandeira PharmaSystem.

Protótipo

```
LONG EXPORT ScopePreAutorizacaoMedicamento (BYTE _TipoConvenio,
                                            BYTE _CodRede,
                                            char *NumCpFiscal)
```

Parâmetros

[in]	BYTE	_TipoConvenio	Tipo do convênio. (0=PBM, 1=Empresa)
[in]	BYTE	_CodRede	Código da rede (ver Convênios)

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
BYTE    TipoConvenio, CodigoRede;
...
// abre sessão
...
// coleta o tipo do convênio, o código da rede
...
```

```
retorno = ScopePreAutorizacaoMedicamento (TipoConvenio, CodigoRede);
...
// processa a transação
...
// fecha a sessão
...
```

Cancelamento de Pre-Autorizacao de Medicamentos PBM

Para cancelar um produto da lista dos produtos autorizados, deve-se chamar a função ScopeCancelaPreAutMedicamento(), que excluirá o produto e a quantidade da lista de produtos. A função pode retornar no fluxo de coleta o código de coleta 64579 para fornecer a lista de medicamentos, e o código de coleta 64580 para obter a lista de medicamentos. Essa função suporta a bandeira PharmaSystem.

Protótipo

```
LONG EXPORT ScopeCancelaPreAutMedicamento()
```

Parâmetros

[in]	BYTE	_TipoConvenio	Tipo do convênio. (0=PBM, 1=Empresa)
[in]	BYTE	_CodRede	Código da rede (ver Convênios)

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Exemplo

```
...
BYTE    TipoConvenio, CodigoRede;
...
// abre sessão
...
// coleta o tipo do convênio, o código da rede
...
retorno = ScopeCancelaPreAutMedicamento (TipoConvenio, CodigoRede);
...
// processa a transação
...
// fecha a sessão
...
```

Funções de PIN-Pad

Esta seção documenta as funções de acesso ao PIN-Pad utilizando a biblioteca compartilhada. Estas funções seguem o formato ScopePP*Funcao*(). Utilize o apêndice B, C e a tabela de código de retorno como complemento.

IMPORTANTE: A utilização destas funções, com exceção das usadas para a exibição de mensagens no visor, está condicionada à configuração no ScopeCNF, onde a opção "*Uso exclusivo do Scope*" deve estar desmarcada.

Comunicação com o PIN-Pad

Como em qualquer dispositivo, a comunicação com o PIN-Pad precisa ser iniciada e, após o uso, finalizada: ScopePPOpen() e ScopePPClose(). As funções aqui relacionadas devem sempre ser chamadas quando a opção *"Uso exclusivo do Scope"* não estiver marcada.

Iniciando a comunicação

Antes do início de qualquer transação, a aplicação deve abrir o PIN-Pad, ou seja, iniciar o canal de comunicação com o PIN-Pad. A chamada bem sucedida desta função é pré-requisito para todas as outras da interface com o PIN-Pad.

Protótipo

```
LONG EXPORT ScopePPOpen (WORD PortaSerial)
```

Parâmetros

[in]	WORD	PortaSerial	Porta serial que se encontra conectado o PIN-Pad.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x000E	14	PIN-Pad já foi aberto
0x001F	31	Erro de comunicação: PIN-Pad provavelmente desconectado ou problemas com a interface serial. Pode ser que a porta que está passando como parâmetro esteja errada.
0x00CA	202	Memória não alocada para a estrutura do PIN-Pad compartilhado
0x00CB	203	Erro alocando memória
0x00CD	205	PIN-Pad já aberto via SCOPE

Exemplo

```
...
WORD porta = 1;
LONG retorno = 0;
...
```

```
retorno = ScopePPOpen(porta);
...
```

Encerrando a comunicação

Uma vez que o PIN-Pad não será mais utilizado, a aplicação pode encerrar a comunicação com o PIN-Pad, deixando uma mensagem no visor (*display*) do PIN-Pad.

Protótipo

```
LONG EXPORT ScopePPClose (char* IdleMsg)
```

Parâmetros

[in]	String	IdleMsg	Mensagem de 32 caracteres (2 linhas x 16 colunas) a ser mantida no display do PIN-Pad após o fechamento

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x00D3	211	PIN-Pad não foi aberto pela aplicação

Exemplo

```
...
LONG retorno = 0;
...
retorno = ScopePPClose("  Itautec S.A.  ");
...
```

Mensagens no visor

Para personalização da aplicação de frente de loja, a aplicação pode enviar mensagens para o PIN-Pad, desde que o SCOPE Client não esteja processando alguma transação.

IMPORTANTE: Estas são as únicas funções que não dependem da configuração do ScopeCNF quanto ao uso exclusivo do SCOPE.

Mensagens de tamanho padrão

Devido à diversidade de formatos de visor, por padrão, todos os PIN-Pads devem prover um visor com o mínimo de 2 linhas por 16 colunas para exibir mensagens. Para este formato padrão existe a função ScopePPDisplay().

Protótipo

```
LONG EXPORT ScopePPDisplay (char* Msg)
```

Parâmetros

[in]	String	Msg	Mensagem de 32 caracteres (2 linhas x 16 colunas) a ser apresentada no display do PIN-Pad.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x00D1	209	PIN-Pad não configurado
0x00D2	210	Display não permitido neste momento ou situação
0x00D3	211	PIN-Pad não foi aberto pela aplicação

Exemplo

```
...
LONG retorno = 0;
...
retorno = ScopePPDisplay("  Itautec S.A.  ");
...
```

Mensagens maiores

Para aproveitar os recursos do visor de certos equipamentos, o SCOPE Client disponibiliza a função ScopePPDisplayEx() que recebe, junto com a mensagem, o tamanho desta. O parâmetro é uma String que obedece ao formato abaixo:

Posição	Formato	Descrição
001-003	N3	Tamanho da mensagem a seguir (xxx).
004-???	AXXX	Mensagem a ser apresentada, podendo conter caracteres de controle aceitos pelo display do PIN-Pad, tal como o CR (0Dh) para quebra de linha

Protótipo

```
LONG EXPORT ScopePPDisplayEx (char* Msg)
```

Parâmetros

[in]	String	Msg	String ASCII de caracteres com o formato abaixo.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x00D1	209	PIN-Pad não configurado
0x00D2	210	Display não permitido neste momento ou situação
0x00D3	211	PIN-Pad não foi aberto pela aplicação

Exemplo

```
...
```

```
char *mensagem = "065Enviando uma mensagem maior que a mensagem padrão para
o PIN-Pad";
LONG retorno = 0;
...
retorno = ScopePPDisplayEx(mensagem);
...
```

Informações obtidas do PIN-Pad

Internamente, os PIN-Pads com a biblioteca compartilhada, possuem informações armazenadas sobre a versão dos dados das redes autorizadoras e do próprio dispositivo. Quando o parque de PDVs é muito grande, o levantamento via aplicativo dos PIN-Pads é uma maneira mais segura e rápida que o método manual.

Dados das redes autorizadoras

Para algumas autorizadoras, o PIN-Pad armazena tabelas com dados para utilização nas leituras de cartão e senha. Com o tempo, estes dados podem ficar obsoletos, sendo necessário o armazenamento da versão destas tabelas nos PIN-Pads. Estas tabelas são retornadas pelo PIN-Pad através da função ScopePPGetInfo(), no formato abaixo.

Posição	Formato	Descrição
001-020	A20	Nome da rede adquirente (com espaços à direita)
021-033	A13	Versão da aplicação da rede adquirente, no formato "VVV.VV AAMMDD"
034-040	A7	Informações proprietárias da rede adquirente
041-042	A2	Tamanho em bytes dos dados a seguir ("00" a "yy")
043-???	Hxx(Byy)	Dados binários de identificação do SAM, caso existente, no layout exigido pela rede adquirente

Dados do dispositivo

O formato das informações do PIN-Pad é padrão e segue o formato abaixo:

Posição	Formato	Descrição
001-020	A20	Nome do fabricante do PIN-Pad
021-040	A20	Modelo / versão do hardware, no formato "xxx...xxx;mmm", onde "xxxx" é o Nome do equipamento e "mmm" a capacidade de memória ("512KB", "1MB", "2MB", ...)
041-060	A20	Versão do firmware (formato livre)
061-064	A4	Versão da especificação, no formato "V.VV" (neste caso, fixo em "1.05")
065-080	A16	Versão da aplicação básica, no formato "VVV.VV AAMMDD" (com 3 espaços à direita)
081-100	A20	Número de série do PIN-Pad (com espaços à direita)

Obtendo as informações

A função ScopePPGetInfo() retorna as informações sobre o PIN-Pad e suas aplicações, como descrito acima. Caso alguma informação não exista ou não se aplique para o modelo de PIN-Pad, ela deverá ser fornecida em branco (espaços). Para decidir qual informação a aplicação quer obter, o primeiro parâmetro deve receber o valor correspondente:

- valor 0: dados do PIN-Pad;
- valor maior que 0: dependendo do valor, serão obtidas as informações específicas das aplicações das autorizadoras.

Protótipo

```
LONG EXPORT ScopePPGetInfo (WORD IdSaida, WORD DadosLen, char* Dados)
```

Parâmetros

[in]	WORD	IdSaida	Informa a opção do formato dos dados de saída.
[in]	WORD	DadosLen	Tamanho reservado pelo aplicativo para receber os dados.
[out]	String	Dados	Dados sobre o PIN-Pad. Utilize as tabelas descritas nas sessões acima para a especificação do buffer retornado.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x00CA	202	Memória não alocada para a estrutura do PIN-Pad compartilhado
0x00D1	209	PIN-Pad não configurado
0x00D2	210	Display não permitido neste momento ou situação
0x00D3	211	PIN-Pad não foi aberto pela aplicação

Exemplo

```
...
char aux[255];
...
if (ScopePPGetInfo(0, (WORD) sizeof(aux), aux))
{
   printf("Nome do fabricante do PIN-Pad..: [%20.20s]\n", &aux[00]);
   printf("Modelo / versao do hardware....: [%20.20s]\n", &aux[20]);
   printf("Versao do firmware.............: [%20.20s]\n", &aux[40]);
   printf("Versao da especificacao........: [%4.4s]  \n", &aux[60]);
   printf("Versao da aplicacao basica.....: [%16.16s]\n", &aux[64]);
   printf("Numero de serie do PIN-Pad.....: [%20.20s]\n", &aux[80]);
}
...
```

Obtendo teclas do PIN-Pad

O par de funções ScopePPStartGetKey() / ScopePPGetKey() captura uma tecla pressionada no PIN-Pad. Elas fazem parte do conjunto de funções não-blocantes. Não-blocantes significa que a chamada a essas funções são de retorno imediato, não interrompendo o fluxo de processamento da aplicação.

Iniciando a leitura da tecla

A função ScopePPStartGetKey() é responsável pelo início do processo, colocando o PIN-Pad em modo de espera por uma tecla não-numérica.

Protótipo

```
LONG EXPORT ScopePPStartGetKey (void)
```

Parâmetros

Não há parâmetro.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x000A	10	Necessária operação previa
0x000F	15	PIN-Pad não foi aberto
0x00D3	211	PIN-Pad não foi aberto pela aplicação

Obtendo a tecla

Após ter posto o PIN-Pad em modo de captura de tecla, a função ScopePPStartGetKey() deverá ser chamada continuamente até que o retorno dela seja diferente de 1 (PIN-Pad em processamento). Esta função finaliza o processo iniciado por ScopePPStartGetKey(). Enquanto o retorno for igual a 1, o processo pode ser cancelado pelo *checkout* através da função ScopePPAbort().

Protótipo

```
LONG EXPORT ScopePPGetKey (void)
```

Parâmetros

Não há parâmetro.

Retorno

0 Pressionada tecla de confirmação (OK ou ENTER)
4 Pressionada tecla de função #1.
5 Pressionada tecla de função #2.
6 Pressionada tecla de função #3.
7 Pressionada tecla de função #4.
8 Pressionada tecla de limpeza (backspace).
13 Pressionada tecla de cancelamento.

Para os valores de retorno, veja a tabela de código de retorno.

Exemplo

```
...
LONG retorno = 0;
```

```
...
if (ScopePPStartGetKey() == 0) // se OK
{
   printf("\nAguardando tecla a ser pressionada no PIN-Pad...");
   do
   {
      retorno = ScopePPGetKey();
   }
   while (retorno == 1);   // enquanto em processamento
   switch (retorno)
   {
   case 0: printf("\nPressionada a tecla <OK>\n"); break;
   case 4: printf("\nPressionada a tecla de funcao 1\n"); break;
   case 5: printf("\nPressionada a tecla de funcao 2\n"); break;
   case 6: printf("\nPressionada a tecla de funcao 3\n"); break;
   case 7: printf("\nPressionada a tecla de funcao 4\n"); break;
   case 8: printf("\nPressionada a tecla <Limpa>\n"); break;
   case 13: printf("\nPressionada a tecla <Cancela>\n"); break;
   default: printf("\nERRO: = %d\n", retorno); break;
   }
}
...
```

Obtendo a senha aberta

O conjunto ScopePPStartGetPIN() / ScopePPGetPIN(), também não-blocantes, é responsável pela captura da senha do usuário. A senha é criptografada pelo PIN-Pad com a *Master Key* da Itautec, e será retornada sem criptografia pela função ScopePPGetPIN(), em caso de sucesso. A *Master Key* da Itautec está inserida, por padrão, nos PIN-Pads homologados pela Visanet.

CUIDADO: A utilização destas funções é apenas para a captura de informações que diz respeito apenas ao próprio estabelecimento (exemplo: senha do supervisor da loja ou de algum funcionário) e não deve ser utilizado para captura de senha de cartões de banco dos clientes.

Como a informação digitada no PIN-Pad é aberta e é possível passar a mensagem a ser exibida no display do PIN-Pad, a aplicação pode utilizá-las para coletar apenas informações de domínio da aplicação.

Iniciando o modo de leitura da senha aberta

A função ScopePPStartGetPIN() é responsável pelo início do processo, o qual é complementado pela função ScopePPGetPIN().

Protótipo

```
LONG EXPORT ScopePPStartGetPIN (char *MsgDisplay)
```

Parâmetros

[in]	String	MsgDisplay	Mensagem que será exibida no PIN-Pad. Esta mensagem deve ter no máximo 32 caracteres sendo duas linhas de 16 caracteres.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x00CE	206	Não foi possível definir a *Master Key* a ser utilizada
0x000F	15	PIN-Pad não foi aberto

Lendo a senha aberta

O processo de leitura da senha é finalizado por ScopePPGetPIN(). Ela deve ser chamada constantemente enquanto retornar o valor igual a 1 (PIN-Pad em processamento). Nessa situação, o processo de captura de senha pode ser cancelado pelo *checkout* através da função ScopePPAbort().

Protótipo

```
LONG EXPORT ScopePPGetPIN (char* PIN)
```

Parâmetros

[out]	String	PIN	Senha capturada do usuário (já descriptografada), finalizada com caractere nulo ('\0').

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x00D0	208	Erro no parâmetro da função GetPIN

Exemplo

```
...
char pin[40];
LONG retorno = 0;
...
if (ScopePPStartGetPIN("DIGITE A SENHA") == 0)
{
   printf("\nAguardando digitação da senha...");
   do
   {
      retorno = ScopePPGetPIN(pin);
   } while (retorno == 1); // enquanto em processamento
   if (retorno == 0)       // se OK
```

```
        printf("\nA senha digitada é [%s]\n", pin);
}
...
```

Obtendo a senha criptografada

O conjunto ScopePPStartGetPINEx()/ScopePPGetPINEx() é responsável pela captura da senha do usuário. Estas funções são úteis para clientes que possuem algum módulo de descriptografia e vai trafegar a informação, que é sigilosa, por algum meio condutor.

A senha é criptografada pelo PIN-Pad com a *Master Key* da Itautec e será retornada com criptografia pelo método ScopePPGetPINEx(). A *Master Key* da Itautec está inserida, por padrão, nos PIN-Pads homologados pela Visanet.

Iniciando o modo de leitura da senha criptografada

A função ScopePPStartGetPINEx() é responsável pelo início do processo, o qual é complementado pela função ScopePPGetPINEx().

Protótipo

```
LONG EXPORT ScopePPStartGetPINEx (char* msgDisplay, int mode, int mkey,
char* wkey, char* pan)
```

Parâmetros

[in]	String	msgDisplay	Mensagem que será exibida no PIN-Pad. Esta mensagem deve ter no máximo 32 caracteres sendo duas linhas de 16 caracteres.
[in]	int	mode	Modo de criptografia. Previsto receber só o valor 0, que usa a *Master Key* / *Working Key* (DES). Outros modos serão utilizados no futuro.
[in]	int	mkey	Índice da *MasterKey*. Previsto receber apenas o valor 0. Outros índices para uso futuro.
[in]	String	wkey	A *Working Key*. Para o modo 0, deverão ser enviados 8 bytes. Caso seja *null*, a função não devolverá a senha criptografada e o PAN não será utilizado.
[in]	String	pan	PAN do cartão com tamanho entre 11 e 19 dígitos. Caso o pan ou a wkey seja *null*, será utilizado um valor fixo interno.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x0002	2	Notificar o operador com a mensagem recebida do PIN-Pad
0x000A	10	Necessária operação previa
0x000F	15	PIN-Pad não foi aberto
0x0010	16	Erro interno de execução
0x00CE	206	Não foi possível definir a *Master Key* a ser utilizada
0x00D3	211	PIN-Pad não foi aberto pela aplicação

Lendo a senha criptografada

A função ScopePPGetPINEx() finaliza o processo iniciado por ScopePPStartGetPINEx(). Ela deve ser chamada constantemente enquanto retornar valor igual a 1 (PIN-Pad em processamento). Nessa situação, o processo de captura de senha pode ser cancelado através da função ScopePPAbort(). Em caso de sucesso, esta função retorna o valor 0. Neste caso, a senha criptografada de acordo com os parâmetros usados na função ScopePPStartGetPINEx() é retornada no parâmetro PIN.

Protótipo

```
LONG EXPORT ScopePPGetPINEx (char *PIN)
```

Parâmetros

[in]	String	PIN	Senha retornada pelo PIN-Pad, criptografada. Para senhas criptografadas com os algoritmos DES/3DES, o PIN deve ter 16 bytes. Para senhas criptografadas com o algoritmo DUKPT, PIN dever ter 36 bytes.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x00D0	208	Erro no parâmetro da função GetPIN
0x00D3	211	PIN-Pad não foi aberto pela aplicação

Exemplo

```
...
char pin[40];
LONG retorno = 0;
...
if (ScopePPStartGetPINEx("DIGITE A SENHA", 0, 0, NULL, NULL) == 0)
{
   printf("\nAguardando digitação da senha...");
   do
   {
      retorno = ScopePPGetPINEx(pin);
   } while (retorno == 1); // enquanto em processamento
   if (retorno == 0)       // se OK
      printf("\nA senha digitada é [%s]\n", pin);
}
...
```

Lendo um cartão

Como a leitura de senha e de teclas, a captura de cartão também se dá com o uso de funções não-blocantes: ScopePPStartGetCard() / ScopePPGetCard().

Iniciando a leitura do cartão

ScopePPStartGetCard() é a função que inicia o processo de leitura do cartão, seja ele magnético ou com chip.

Protótipo

```
LONG EXPORT ScopePPStartGetCard (WORD TipoApl, char* ValorInicial)
```

Parâmetros

[in]	WORD	TipoApl	Identifica o tipo de aplicação desejada (crédito = 1, débito = 2 e para qualquer aplicação = 99).
[in]	String	ValorInicial	Valor inicial da transação. Podendo ser 0 (zero) caso este dado não esteja disponível. Este campo deverá ter tamanho máximo de 12 (doze) bytes, sendo 2 (duas) casas decimais.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x00D3	211	PIN-Pad não foi aberto pela aplicação

Recuperando o cartão

A função ScopePPGetCard() finaliza o processo iniciado por ScopePPStartGetCard() e deve ser chamada diversas vezes enquanto retornar 1 (PIN-Pad em processamento) ou 2 (exibir para o cliente e para o operador a mensagem que o PIN-Pad passou). Enquanto nessa situação, o processo pode ser cancelado pelo *checkout* através da função ScopePPAbort().

Quando a função ScopePPGetCard() retornar o código igual a 2, o PIN-Pad colocou uma mensagem no segundo parâmetro da função e a aplicação deve exibi-la na tela. Logo após a exibição da mensagem, a aplicação deve continuar chamando esta função até que esta não retorne mais um dos dois valores relacionados acima. Ao retornar o valor 0 (sucesso), o último parâmetro da função estará preenchido com os dados do cartão que segue o seguinte formato:

Posição	Formato	Descrição
001-002	N2	Tipo de cartão lido: • "00" – Magnético • "01" - Moedeiro VisaCash sobre TIBC v1 • "02" - Moedeiro VisaCash sobre TIBC v3 • "03" – EMV • "04" - Easy-Entry sobre TIBC v1
003-004	N2	Tamanho da trilha 1

005-080	A76	Trilha 1 (sem as sentinelas e com o byte de formato - primeiro caractere alfanumérico), alinhada à esquerda com espaços à direita.
081-082	N2	Tamanho da trilha 2
083-119	A37	Trilha 2 (sem as sentinelas), alinhada à esquerda com espaços à direita.
120-121	N2	Tamanho do PAN
122-139	A19	PAN, alinhado à esquerda com espaços à direita
140-166	A26	Nome do proprietário do cartão, com espaços à direita

Protótipo

```
LONG EXPORT ScopePPGetCard (WORD Id, char* MsgNotify, WORD Len, char* Dados)
```

Parâmetros

[in]	WORD	Id	Informa a opção do formato dos dados de saída. Inicialmente fixo com 0 (zero).
[in]	String	MsgNotify	Mensagem de 32 caracteres a ser apresentada no "checkout" caso a função retorne PP_NOTIFY.
[in]	WORD	Len	Tamanho do buffer alocado para os dados.
[out]	String	Dados	Dados sobre o cartão finalizado. Consulte a tabela abaixo para a especificação do formato.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x0002	2	Notificar o operador com a mensagem recebida do PIN-Pad
0x000B	11	Parâmetro inválido
0x00D3	211	PIN-Pad não foi aberto pela aplicação

Exemplo

```
...
char cartao [256];
char mensagem [256];
LONG retorno = 0;
...
if (ScopePPStartGetCard(99, "000") == 0)
{
   printf("\nAguardando inserir ou passar o cartao...");
   do
   {
      retorno = ScopePPGetCard(0, mensagem, sizeof(cartao), cartao);
      if (retorno == 2)
         printf("\nMensagem do PIN-Pad\n[%s]\n", mensagem);
   } while (retorno == 1); // enquanto em processamento
   if (retorno == 0)       // se OK
```

```
      printf("\nOs dados do cartão são [%s]\n", cartao);
}
...
```

Abortando o processamento (*ScopePPAbort*)

Esta função finaliza um processo em andamento, iniciado por uma função do tipo não-blocante que seguem o padrão ScopePPStart*Func*(). Ela deve ser chamada caso o operador do *checkout* decida abortar a operação enquanto a função complementar ScopePP*Func*() estiver retornando o valor 1 (PIN-Pad em processamento).

Protótipo

```
LONG EXPORT ScopePPAbort (void)
```

Parâmetros

Não há parâmetro.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x00D3	211	PIN-Pad não foi aberto pela aplicação

Exemplo

```
...
char cartao [256];
char mensagem [256];
LONG retorno = 0;
...
if (ScopePPStartGetCard(99, "000") == 0)
{
   printf("\nAguardando inserir ou passar o cartao...");
   do
   {
      retorno = ScopePPGetCard(0, mensagem, sizeof(cartao), cartao);
      if (retorno == 2)
         printf("\nMensagem do PIN-Pad\n[%s]\n", mensagem);
      if (_kbhit())  // se operador pressionou tecla do teclado
      {      // cancela o processamento do cartao
         ScopePPAbort();
         retorno = 13;
      }
   } while (retorno == 1 || retorno == 2); // enquanto em processamento ou
mensagem
```

```
   if (retorno == 0)       // se OK
      printf("\nOs dados do cartão são [%s]\n", cartao);
}
...
```

Mensagens de erro (*ScopePPMsgErro*)

Esta função retorna a descrição referente ao código de erro informado retornado pelas funções de acesso ao PIN-Pad.

Protótipo

```
LONG EXPORT ScopePPMsgErro (LONG RC, char *MsgErro)
```

Parâmetros

[in]	WORD	RC	Código do erro.
[in]	String	MsgErro	Mensagem referente ao código de erro informado.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x00D3	211	PIN-Pad não foi aberto pela aplicação

Exemplo

```
...
char cartao [256];
char mensagem [256];
LONG retorno ;
...
retorno = ScopePPStartGetCard(99, "000");
if (retorno == 0)
{
   printf("\nAguardando inserir ou passar o cartao...");
   do
   {
      retorno = ScopePPGetCard(0, mensagem, sizeof(cartao), cartao);
      if (retorno == 2)
         printf("\nMensagem do PIN-Pad\n[%s]\n", mensagem);
      if (_kbhit())  // se operador pressionou tecla do teclado
      {       // cancela o processamento do cartao
         ScopePPAbort();
         retorno = 13;
      }
```

```
   } while (retorno == 1 || retorno == 2); // enquanto em processamento
                           // ou mensagem
}
...
if (retorno == 0)          // se OK
   printf("\nOs dados do cartão são [%s]\n", cartao);
else
{
   ScopePPMsgErro (retorno, mensagem);
   printf("\n[%s]\n", mensagem);
}
...
```

Obtendo a comanda de um cartão

A leitura de um cartão do tipo comanda se dá com o uso de funções não-blocantes: ScopePPStartObtemComanda() e, para o tipo comanda café, ScopePPObtemComanda().

Iniciando o modo de leitura da comanda

ScopePPStartObtemComanda () é a função que inicia o processo de leitura do cartão, inicializa o *timer* a ser utilizado para verificação de *time out*, e direciona para a função que trata aquele determinado tipo de comanda.

Protótipo

```
LONG EXPORT ScopePPStartObtemComanda (int  TipoComanda)
```

Parâmetros

[in]	int	TipoComanda	Tipo de comanda (Atualmente existe apenas o tipo Comanda Café)

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x000A	10	Necessária operação previa
0x000F	15	PIN-Pad não foi aberto
0x00CA	202	Memória não alocada para a estrutura do PIN-Pad compartilhado
0x00D3	211	PIN-Pad não foi aberto pela aplicação

Lendo a comanda

A função ScopePPObtemComanda() finaliza o processo iniciado por ScopePPStartObtemComanda(). É chamada caso o tipo seja comanda café, onde o cartão tem chip. Ela deve ser chamada constantemente enquanto retornar valor igual a 1 (PIN-Pad em processamento). Em caso de sucesso, esta função retorna o valor 0. Neste caso, a comanda será salva na área de memória passada como parâmetro. Existe a opção de permanecer com o cartão no PIN-Pad, proporcionando a limpeza do cartão logo depois da leitura, sem necessidade de chamada separada dessa função. Para essa opção, deve ser passado o valor '0' no parâmetro RemoveComanda.

Protótipo

```
LONG EXPORT ScopePPObtemComanda(int TipoComanda, char *Comanda, int RemoveComanda);
```

Parâmetros

[in]	int	TipoComanda	Tipo de comanda (Atualmente existe apenas o tipo Comanda Café)
[in]	char	*Comanda	Buffer onde será salva a comanda lida nessa operação
[in]	int	RemoveComanda	Opção de remover a comanda posteriormente à leitura (1), ou permanecer com o cartão para posterior limpeza (0).

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x000D	13	Operação cancelada
0x0021	33	Mensagem recebida do PIN-Pad possui formato inválido.
0x00D3	211	PIN-Pad não foi aberto pela aplicação
0x00D4	212	Time-out do cliente / usuário
0x00D6	214	Comanda Vazia
0X00D7	215	Comanda Inválida

Exemplo

```
...
char szComanda[4096+1];
LONG Retorno;


...
Retorno = ScopePPStartObtemComanda (comanda_cafe);


if (Retorno == 0)
{
   printf("\nCOMANDA: INSIRA O CARTAO\n");
}
```

```
do
{
memset(szComanda, 0x00, sizeof(szComanda));
Retorno = ScopePPObtemComanda(comanda_cafe, szComanda, 1);
}
while (Retorno == 1);
...
```

Limpando a comanda de um cartão

A limpeza de um cartão do tipo comanda se dá com o uso de funções não-blocantes: ScopePPStartLimpaComanda () e, para o tipo comanda café, ScopePPLimpaComanda ().

Iniciando o modo de limpeza da comanda

ScopePPStartLimpaComanda() é a função que inicia o processo de limpeza do cartão, inicializa o *timer* a ser utilizado para verificação de *time out*, e direciona para a função que trata aquele determinado tipo de comanda.

Protótipo

```
LONG EXPORT ScopePPStartLimpaComanda (int  TipoComanda)
```

Parâmetros

[in]	int	TipoComanda	Tipo de comanda (Atualmente existe apenas o tipo Comanda Café)

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x00D3	211	PIN-Pad não foi aberto pela aplicação
0X00D7	215	Comanda Inválida

Limpando a comanda

A função ScopePPLimpaComanda() finaliza o processo iniciado por ScopePPStartLimpaComanda(). É chamada caso o tipo seja comanda café, onde o cartão tem chip. Ela deve ser chamada constantemente enquanto retornar valor igual a 1 (PIN-Pad em processamento). Em caso de sucesso, esta função retorna o valor 0.

Protótipo

```
LONG EXPORT ScopePPLimpaComanda(int  TipoComanda);
```

Parâmetros

[in]	int	TipoComanda	Tipo de comanda (Atualmente existe apenas o tipo Comanda Café)

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	**Decimal**	
0x000D	13	Operação cancelada
0x0021	33	Mensagem recebida do PIN-Pad possui formato inválido.
0x00D3	211	PIN-Pad não foi aberto pela aplicação
0x00D4	212	Time-out do cliente / usuário
0x00D6	214	Comanda Vazia
0X00D7	215	Comanda Inválida

Exemplo

```
...
LONG Retorno;
...
Retorno = ScopePPStartLimpaComanda (comanda_cafe);
if (Retorno == 0)
{
    printf("\nCOMANDA: INSIRA O CARTAO\n");
}
do
{
    Retorno = ScopePPLimpaComanda(comanda_cafe);
}
while (Retorno == 1);
...
```

Totalização de TEF

Esta sessão trata uma funcionalidade que o SCOPE disponibiliza para a aplicação de frente de loja: a totalização de TEF.

Relatório de TEF

Pensando na facilidade de fazer o balanço dos valores de venda realizados no checkout, o SCOPE Client provê à aplicação um relatório com informações do total de TEF confirmadas e canceladas por bandeira num certo período e que pode ser enviado para a impressora ou para outro meio de armazenamento. Estas informações podem ser levantadas por PDV ou por operador. No entanto, mesmo que "a aplicação se interesse" pelas informações relativas apenas por uma das opções (PDV ou operador), o SCOPE atualizará ambos. Portanto, não haverá otimização ao escolher apenas um dos dois.

No caso do PDV, o período escolhido para a totalização dos valores pode ser diário, semanal, mensal, etc., mas recomenda-se que abranja o período do movimento, isto é, ao iniciar o período num dia, seja zerado o totalizador e, no final do dia, o relatório seja obtido.

Quanto ao relatório por operador, aconselha-se que seja zerado o totalizador na entrada do operador ao caixa e obtenha-se o relatório, na saída do mesmo.

IMPORTANTE: o SCOPE não controla (identifica) qual operador está operando o PDV. É papel da aplicação o controle da entrada e saída de cada operador e consequentemente, iniciar o totalizador e obter o relatório no momento correto.

Iniciando os totais

No início do período de totalização, a aplicação deverá chamar a função ScopeIniciaTotalTEF(), passando no parâmetro o código referente ao totalizador que se deseja zerar, conforme abaixo:

- valor 0: será zerados ambos os totalizadores, o do PDV e o do operador
- valor 1: será zerado apenas o do operador

Protótipo

```
LONG ScopeIniciaTotalTEF (BYTE Nivel)
```

Parâmetros

[in]	BYTE	Nivel	Indica a que nível deverá ser inicializado os totais de TEF (0: ambos, 1: apenas do operador)

Retorno

Código	Descrição	Ação
0x0000	Sucesso na inicialização dos totalizadores	
0xFB07	Erro na totalização de TEF, causado por	- verifique o valor do parâmetro

	um dos motivos: - parâmetro inválido; - não foi possível criar o arquivo; - não foi possível atualizar o arquivo; - não foi possível ler o arquivo.	passado; - verifique o parâmetro passado à função; - verifique direitos de gravação, criação e leitura de arquivos para o usuário logado no PDV; - verifique se existe espaço disponível no disco.
0xFE00	Transação em andamento, o que impossibilita a execução desta função	Revisar a aplicação, pois, ela está chamando a função em momentos errado.

Ver tabela de código de retorno.

Exemplo

```
...
// carrega configuração do PDV de um novo dia
...
// conecta ao SCOPE Server
...
if (ScopeIniciaTotalTEF (0) != 0) // zera ambos
{
   // erro ao zerar o arquivo
}
...
```

Cupom dos totais

Não importa se é por PDV ou por operador, uma forma que o SCOPE entrega o relatório para a aplicação é um texto formatado, com 40 colunas, que pode ser enviado diretamente para a impressora, conforme exemplo abaixo:

```
------        CARTAO DE CREDITO        ----
BANDEIRA              EFETUADO   CANCELADO
VISA                    167.00        0.00
MASTERCARD              564.80        0.00

------        CARTAO DE DEBITO         ----
BANDEIRA              EFETUADO   CANCELADO
CHEQUE ELETRONIC       1650.00        0.00
ELECTRON                 18.00        4.00

------        CARTAO DE DEBITO (CDC)   ----
BANDEIRA              EFETUADO   CANCELADO
```

```
------        CONSULTA CHEQUE          ----
BANDEIRA            CONSULTADO
```

CUIDADO: O valor efetuado contém o valor cancelado. Assim, no exemplo anterior, a bandeira Electron indica uma receita de R$ 14,00 (R$ 18,00 - R$ 4,00) para a loja.

Protótipo

```
LONG ScopeObtemTotalTEF (BYTE Nivel, char *Cupom, WORD TamCupom, char SeparadorLinhasCupom)
```

Parâmetros

[in]	BYTE	Nivel	Indica a que nível deverá ser inicializado os totais de TEF (0: PDV, 1: operador)
[out]	String	Cupom	Cupom com o relatório de totais de TEF por bandeira
[in]	WORD	TamCupom	Tamanho da área que a aplicação alocou pela variável Cupom
[in]	char	SeparadorLinhasCupom	Caractere a ser usado na separação das linhas do cupom. Atualmente, suporta apenas o caractere '@'. Para qualquer outro caractere será utilizado '\n'.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFB07	64263	Erro na totalização de TEF

Exemplo

```
...
WORD TamCupom;
char Cupom[2048];
...
// encerrando o Dia
...
TamCupom = sizeof(Cupom);
if(ScopeObtemTotalTEF(0, Cupom, TamCupom, '\n') == 0)
{
   ...
   EnviaCupomParaImpressora(Cupom);
   ...
}
else
```

```
{
   // erro ao obter o cupom com o relatório
}
...
```

Dados dos totais

A outra maneira que o SCOPE fornece o relatório para a aplicação é por meio de um *buffer* com os campos de tamanho fixo. Há 3 tipos de *layouts* conforme as tabelas abaixo:

Layout Versão 1:

Tamanho	Descrição
3 bytes	Identificação do buffer (“R01”)
2 bytes	Quantidade de bandeiras retornadas que indica a quantidade de repetições dos registros abaixo
1 byte	Identifica a função: - ‘C’: Crédito - ‘D’: Débito; - ‘H’: Cheque - ‘A’: CDC
2 bytes	Código da bandeira
17 bytes	Descrição da bandeira
14 bytes	Valor efetivado (2 casas decimais)
14 bytes	Valor cancelado (2 casas decimais)

Layout Versão 2:

Tamanho	Descrição
3 bytes	Identificação do buffer (“R02”)
3 bytes	Quantidade de bandeiras retornadas que indica a quantidade de repetições dos registros abaixo
3 bytes	Código da bandeira
17 bytes	Descrição da bandeira
1 byte	Quantidade de Funções retornadas que indica a quantidade de repetições dos registros abaixo (29 bytes por função)
1 byte	Identifica a função: - ‘C’: Crédito - ‘D’: Débito; - ‘H’: Cheque - ‘A’: CDC - ‘S’: Saque
14	Valor efetivado (2 casas decimais)

		bytes	
		14 bytes	Valor cancelado (2 casas decimais)

Layout Versão 3:

Tamanho	Descrição
3 bytes	Identificação do buffer (“R03”)
3 bytes	Quantidade de bandeiras retornadas que indica a quantidade de repetições dos registros abaixo
1 byte	Identifica a função: - ‘C’: Crédito - ‘D’: Débito; - ‘H’: Cheque - ‘A’: CDC
3 bytes	Código da bandeira
17 bytes	Descrição da bandeira
14 bytes	Valor efetivado (2 casas decimais)
14 bytes	Valor cancelado (2 casas decimais)

Protótipo

```
LONG ScopeObtemDadosTotalTEF (BYTE Nivel, char *Buffer, WORD TamBuffer)
```

Parâmetros

[in]	BYTE	Nivel	Indica a que nível deverá ser inicializado os totais de TEF (0: PDV, 1: Operador)
[out]	char *	Buffer	Retorna os dados de totais TEF
[in]	WORD	TamBuffer	Tamanho da área que a aplicação alocou pela variável Buffer

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFB07	64263	Erro na totalização de TEF

Exemplo

```
...
WORD TamBuffer;
char Buffer[2048];
...
// encerrando o Dia
```

```
...
TamBuffer = sizeof(Buffer);
if(ScopeObtemDadosTotalTEF(0, Buffer, TamBuffer) == 0)
{
   ...
   GeraRelatorioEmArquivo(Buffer);
   ...
}
else
{
   // erro ao obter o cupom com o relatório
}
...
```

CUIDADO: Esta função somente aceita o *layout* Versão 1.

Protótipo

```
LONG ScopeObtemDadosTotalTEFEx (BYTE Versao, BYTE Nivel, char *Buffer,
WORD TamBuffer)
```

Parâmetros

[in]	BYTE	Versao	Versão do *layout* a ser utilizado: 1, 2 ou 3
[in]	BYTE	Nivel	Indica a que nível deverá ser inicializado os totais de TEF (0: PDV, 1: Operador)
[out]	char *	Buffer	Retorna os dados de totais TEF
[in]	WORD	TamBuffer	Tamanho da área que a aplicação alocou pela variável Buffer

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFB07	64263	Erro na totalização de TEF

Exemplo

```
...
WORD TamBuffer;
char Buffer[2048];
...
// encerrando o Dia
...
TamBuffer = sizeof(Buffer);
if(ScopeObtemDadosTotalTEFEx(3, 0, Buffer, TamBuffer) == 0)
```

```
{
   ...
   GeraRelatorioEmArquivo(Buffer);
   ...
}
else
{
   // erro ao obter o cupom com o relatório
}
...
```

Transação de POS para conciliação

Aqui será tratada uma funcionalidade que guia a coleta de dados de um cupom de TEF.

Descrição da funcionalidade

Essa funcionalidade determina um fluxo de coleta que direcione o operador (a automação comercial) a entrar os dados do cupom de uma transação que foi realizada na máquina de POS. Este recurso foi criado para que a automação cormercial obtivesse os dados do cupom para gravar no arquivo de conciliação do SCOPECON e que tais transações fossem conciliadas. Essa funcionalidade não se comunica com o SCOPE Server e a conexão pode estar offline. No entanto, é necessário que o SCOPE Client tenha feito o logon com o Server, isto é, que a função `ScopeOpen` tenha sido executado com sucesso.

A funcionalidade tinha como foco cartões que o SCOPE só passam em POS, como o CONSTRUCARD e o BNDES. No entanto, quaisquer cartões que passaram pelo POS podem usar esse recurso. Há situações em que a rede autorizadora ou até mesmo o próprio SCOPE Server está offline. Quando acontece isso, normalmente os estabelecimentos passam as transações em POS, que são utilizados como contingência do concentrador de TEF.

Em linhas gerais, o portador do cartão ao pagar sua compra na loja com um dos cartões, o funcionário da loja encaminha-o para o sistema que vai fazer o pagamento. Uma vez paga, o funcionário poderá registrar essa venda no PDV para a futura conciliação com a rede autorizadora.

Os dados da transação serão retornados para a automação comercial pela função `ScopeObtemCampoExt2`. Assim, esses dados poderão ser gravados em arquivo e exportados posteriormente para o módulo SCOPECon. Em outras palavras, os dados da transação para a conciliação não é feita pelo SCOPE. É de responsabilidade da automação comercial a geração do arquivo com os dados da transação. A participação do SCOPE nesse processo está em dois momentos:

1. A coleta dos dados do cupom pelo SCOPE Client para que a automação tenha os dados para o arquivo. Nessa situação o SCOPE serve apenas como um guia ao operador para solicitar as informações do cupom. No final da coleta, a automação solicitará ao SCOPE Client tais dados para o arquivo.
2. A conciliação da transação pelo SCOPECon. O arquivo gerado pela automação comercial será importado por esse módulo.

Transação POS

Essa função inicia um fluxo padrão de coleta do SCOPE para coletar as informações necessárias para que a automação comercial gere um registro do arquivo de conciliação que este seja

futuramente importado pelo SCOPECon. No final desse fluxo a automação comercial poderá obter quase todos os campos necessário utilizando a função `ScopeObtemCampoExt2`.

Na loja, o operador de caixa irá com o portador do cartão a um POS passar o cartão para pagar a venda. O operador retornará ao caixa com o cupom do estabelecimento impresso pelo POS e executará essa transação. Os dados contidos na via do estabelecimento são importantes para alimentar os dados da transação. A não utilização do SCOPE poderá ser por diversas situações. A principal é quando o cartão é CONSTRUCARD ou BNDES, pois eles não são tratados pelo SCOPE. Outra situação é quando a rede autorizadora ou o próprio SCOPE Server estiver offline. Nesse caso, a transação será paga no POS, mas o estabelecimento poderá fazer a conciliação da transação.

Os parâmetros da função não são obrigatórios. Passar valores diferentes do padrão simplificará o fluxo de coleta, não sendo solicitado ao operador. Se esta função vai ser chamada para finalizar a venda que está aberta, a automação poderá passar o valor da venda automaticamente.

IMPORTANTE: Essa transação não exige abertura de sessão. Também ela não tem garantia de queda de energia, ou seja, no caso de interrupção da transação, os dados coletados serão perdidos. Se aberta a sessão de TEF, ela não é considerada no limite de 9 TEFs permitidos na sessão.

Protótipo

```
LONG EXPORT ScopeTransacaoPOS(char *Valor,
                              WORD Rede,
                              WORD Bandeira,
                              WORD Servico)
```

Parâmetros

[in]	String com o máximo de 12 dígitos	Valor	Valor da transação com a vírgula implícita (exemplo: R$ 123,00 = “12300”). Passar string vazia para o SCOPE solicitar. Não pode ser nulo nem string apenas com zeros.
[in]	WORD	Rede	Número positivo que representa código da rede autorizadora segundo o SCOPE. Passar zero para o SCOPE solicitar.
[in]	WORD	Bandeira	Número positivo que representa código da bandeira segundo o SCOPE. Passar zero para o SCOPE solicitar.
[in]	WORD	Servico	Número positivo que representa código do serviço segundo o SCOPE. Passar zero para o SCOPE solicitar.

Retorno

Ver tabela de código de retorno no SCOPE.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFB01	64257	Não foi possível criar a "*thread*" na coleta de dados
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFE00	65024	A transação em andamento – a aplicação deve aguardar

Exemplo

```
...
retorno = ScopeTransacaoPOS("", 0, 0, 0);
...
// processa a transação
...
```

Estados de coleta

A tabela abaixo mostra os estados de coleta que o fluxo poderá retornar, no modo coleta. A maioria dos estados não necessitam tratamentos específicos pela automação. Basta que mostre a mensagem na tela do operador e aguarde a digitação. Quando necessário, está descrito abaixo o tratamento específico que a automação poderá fazer.

ESTADO DE COLETA		DESCRIÇÃO	TRATAMENTO
HEX	DECIMAL		
0xFC87	64647	Exibe menu. Para essa transação, esse estado poderá ser utilizado para coletar dois dados diferentes. Um para a coleta da rede. Outro para a coleta da bandeira.	Obter a lista de opções do menu para a exibição de todas numa única vez. Ver tópico específico.
0xFC09	64521	Coletar se a transação é a vista ou não	Padrão
0xFC0A	64522	Coletar se a transação é parcelada pela administradora ou pelo estabelecimento	Padrão
0xFC0E	64526	Coleta a quantidade de parcelas	Padrão
0xFC34	64564	Coleta o valor da transação	Padrão
0xFCC7	64711	Coleta os 6 primeiros dígitos do cartão	Padrão
0xFC18	64536	Coleta os 4 últimos dígitos do cartão	Padrão
0xFC32	64562	Coleta a data da transação no formato DDMMAA	Padrão
0xFCC8	64712	Coleta o campo DOC (NSU)	Padrão
0xFC3A	64570	Coleta o campo CV (NSU do host)	Padrão

0xFCC9	64713	Coleta o campo AUT (código de autorização)	Padrão
0xFCFD	64765	Coleta em andamento	Aguarda a troca de estado. Não solicita dados ao operador.
0xFCFE	64766	Mostrar informações e retornar fluxo para o cliente SCOPE	Mostrar a mensagem para o operador e devolver o controle para o SCOPE. Não solicita dados ao operador.

Constantes em C dos estados de coleta

Esta tabela mostra as constantes definidas no arquivo de cabeçalho ScopeApi.h. Esse arquivo é disponibilizado junto com o SCOPE Client. Muito útil para automações escritas em linguagem C.

CONSTANTE	CÓDIGO EM HEXA
TC_DECIDE_AVISTA	0xFC09
TC_DECIDE_P_ADM_EST	0xFC0A
TC_QTDE_PARCELAS	0xFC0E
TC_ULTIMOS_DIGITOS	0xFC18
TC_COLETA_DDMMAA	0xFC32
TC_COLETA_VALOR	0xFC34
TC_COLETA_NSU_HOST	0xFC3A
TC_EXIBE_MENU	0xFC87
TC_PRIMEIROS_DIGITOS	0xFCC7
TC_COLETA_CAMPO_AUT	0xFCC8
TC_COLETA_CAMPO_DOC	0xFCC9
TC_COLETA_EM_ANDAMENTO	0xFCFD
TC_INFO_RET_FLUXO	0xFCFE

Redes e bandeiras

O SCOPE Client tem um conjunto mínimo padrão implementado internamente para permitir a transação POS. São garantidas as redes CIELO e REDECARD-L0500, cujos respectivos códigos no SCOPE são 102 e 103. As bandeiras garantidas são VISA, MASTERCARD, ELECTRON, ELO DEBITO, ELO CREDITO, BNDES, CONSTRUCARD. Os serviços são: Débito à Vista, Crédito à Vista, Crédito Parc. Adm, Crédito Parc. Loja, Compra CDC e Crediário. A tabela abaixo mostra o relacionamento que o SCOPE Client mantém internamente.

REDE		BANDEIRA		SERVIÇO	
CÓD	NOME	CÓD	NOME	CÓD	NOME
102	CIELO	1	VISA	9	Crédito à Vista
102	CIELO	1	VISA	27	Crédito Parc. Adm

102	CIELO	1	VISA	28	Crédito Parc. Loja
102	CIELO	2	MASTERCARD	9	Crédito à Vista
102	CIELO	2	MASTERCARD	27	Crédito Parc. Adm
102	CIELO	2	MASTERCARD	28	Crédito Parc. Loja
102	CIELO	8	MAESTRO	6	Débito à Vista
102	CIELO	8	MAESTRO	134	Crediário
102	CIELO	21	ELECTRON	6	Débito à Vista
102	CIELO	21	ELECTRON	134	Crediário
102	CIELO	247	BNDES	6	Débito à Vista
102	CIELO	247	BNDES	134	Crediário
102	CIELO	246	CONSTRUCARD	6	Débito à Vista
102	CIELO	246	CONSTRUCARD	134	Crediário
103	REDECARD-L0500	1	VISA	9	Crédito à Vista
103	REDECARD-L0500	1	VISA	27	Crédito Parc. Adm
103	REDECARD-L0500	1	VISA	28	Crédito Parc. Loja
103	REDECARD-L0500	2	MASTERCARD	9	Crédito à Vista
103	REDECARD-L0500	2	MASTERCARD	27	Crédito Parc. Adm
103	REDECARD-L0500	2	MASTERCARD	28	Crédito Parc. Loja
103	REDECARD-L0500	8	MAESTRO	6	Débito à Vista
103	REDECARD-L0500	8	MAESTRO	45	Compra CDC
103	REDECARD-L0500	21	ELECTRON	6	Débito à Vista
103	REDECARD-L0500	21	ELECTRON	45	Compra CDC
103	REDECARD-L0500	246	CONSTRUCARD	6	Débito à Vista
103	REDECARD-L0500	246	CONSTRUCARD	45	Compra CDC

Este relacionamento interno poderá ser desconsiderado se configurado no scope.ini um novo relacionamento. Ver tópico **Configuração do arquivo scope.ini**.

Exemplos de configuração 1

Esse exemplo mostra uma configuração em que a transação teria um comportamento similar como à situação em que o arquivo não está configurado. O comportamento é similar, mas não igual, pois não temos a associação de serviço para cada configuração de rede ou bandeira. Logo, no fluxo as bandeiras não serão separadas conforme a opção escolhida de crédito, débito ou CDC (crediário) e todas elas aparecerão para que o operador escolha uma.

```
[SCOPEAPIPOS]
DigitacaoCodigo=n
Complementar=n
Redes=102,103
Bandeiras=1,2,8,21,246,247
NomeBandeira001=VISA
NomeBandeira002=MASTERCARD
NomeBandeira008=MAESTRO
NomeBandeira021=ELECTRON
NomeBandeira246=BNDES
NomeBandeira247=CONSTRUCARD

[SCOPEAPIPOS-R102]
Nome=CIELO
```

```
Bandeiras=1,2,8,21,247,246

[SCOPEAPIPOS-R103]
Nome= REDECARD
Bandeiras=1,2,8,21, 246
```

Exemplos de configuração 2

Aqui temos uma configuração para um estabelecimento que tem apenas POS da CIELO como contingência. Não será perguntada pela rede REDECARD.

```
[SCOPEAPIPOS]
DigitacaoCodigo=n
Complementar=n
Redes=102
Bandeiras=1,2,8,21,246,247
NomeBandeira001=VISA
NomeBandeira002=MASTERCARD
NomeBandeira008=MAESTRO
NomeBandeira021=ELECTRON
NomeBandeira246=BNDES
NomeBandeira247=CONSTRUCARD

[SCOPEAPIPOS-R102]
Nome=CIELO
Bandeiras=1,2,8,21,247,246
```

Exemplos de configuração 3

A configuração desse exemplo agrega à rede GETNET e as bandeiras SIMCRED e VISA para um estabelecimento que tem um POS da GETNET como contingência além do POS da CIELO e da REDECARD.

```
[SCOPEAPIPOS]
DigitacaoCodigo=n
Complementar=s
Redes=73
Bandeiras=136
NomeBandeira136=SIMCRED

[SCOPEAPIPOS-R073]
Nome=GETNET
Bandeiras=136,1
```

Exemplos de configuração 4

Com essa configuração será dada a opção OUTRA e coletado o código de bandeira se essa opção for selecionada.

```
[SCOPEAPIPOS]
DigitacaoCodigo=s
```

Exemplos de configuração 5

A configuração desse exemplo agrega à rede GETNET e as bandeiras SIMCRED e VISA e ELECTRON para um estabelecimento que tem um POS da GETNET como contingência além do POS da CIELO e da REDECARD. Além disso, associa os serviços de crédito para SIMCRED e VISA e o serviço de débito para ELECTRON.

```
[SCOPEAPIPOS]
DigitacaoCodigo=n
Complementar=s
Redes=73
Bandeiras=136
NomeBandeira136=SIMCRED

[SCOPEAPIPOS-R073]
Nome=GETNET
Bandeiras=136,1,21
ServicoB001=9,027,28
ServicoB136=009,27,028
ServicoB021=6
```

Exemplos de configuração 6

Esse exemplo mostra uma configuração em que a transação teria um comportamento igual à situação em que o arquivo não está configurado. Diferente do exemplo 1, aqui o comportamento é igual, pois temos a associação de serviço para cada configuração de rede ou bandeira.

```
[SCOPEAPIPOS]
DigitacaoCodigo=n
Complementar=n
Redes=102,103
Bandeiras=1,2,8,21,246,247
NomeBandeira001=VISA
NomeBandeira002=MASTERCARD
NomeBandeira008=MAESTRO
NomeBandeira021=ELECTRON
NomeBandeira246=BNDES
NomeBandeira247=CONSTRUCARD

[SCOPEAPIPOS-R102]
Nome=CIELO
Bandeiras=1,2,8,21,247,246
ServicoB001=009,27,28
ServicoB002=9,27,28
ServicoB008=6,134
ServicoB021=6,134
ServicoB247=6,134
ServicoB246=6,134

[SCOPEAPIPOS-R103]
Nome= REDECARD
```

```
Bandeiras=1,2,8,21, 246
ServicoB001=9,27,028
ServicoB002=9,027,28
ServicoB008=6,45
ServicoB021=6,45
ServicoB246=6,45
```

Códigos de serviços

No SCOPE existem vários códigos de serviços disponíveis, mas nem todos são aplicáveis nesta funcionalidade. Cada serviço é agrupado em um grupo de serviço. Abaixo está a relação de grupos de serviços e os serviços previstos como é no SCOPE.

Grupo de serviço		Serviço	
Código	**Descrição**	**Código**	**Descrição**
01	Cartão de Débito	6	Débito à Vista
		45	Compra CDC
		134	Crediário
02	Cartão de Crédito	9	Crédito à Vista
		27	Crédito Parc. Adm
		28	Crédito Parc. Loja

Percebe-se que o CDC e o Crediário estão agrupados em débito, mas para fins da transação POS, os serviços estarão agrupados nos conforme abaixo:

Grupo de serviço	Serviço	
Descrição	**Código**	**Descrição**
Cartão de Débito	6	Débito à Vista
	45	Compra CDC
	134	Crediário
CDC	45	Compra CDC
	134	Crediário
Cartão de Crédito	9	Crédito à Vista
	27	Crédito Parc. Adm
	28	Crédito Parc. Loja

O serviço 134 (crediário) é idêntico ao CDC, mas é utilizado apenas para a rede CIELO. Qualquer outra rede o serviço utilizado deve ser o 45 (compra CDC). Esses dois serviços estão em ambos os grupos, para manter como é o agrupamento no SCOPE. No entanto, escolhendo a opção de CDC logo no começo, simplifica o fluxo, já que não perguntará se é à vista e perguntará o número de parcelas. Consequentemente, o fluxo poderá chegar ao serviço CDC, respondendo tanto débito, quanto CDC na primeira coleta.

Figura 8: cupom da CIELO

Cupom CIELO X ScopeObtemCampoExt2 X arquivo

Nesse tópico são mapeados os campos de cupons da rede CIELO com as constantes da função `ScopeObtemCampoExt2` do SCOPE Client e os campos do registro do arquivo de conciliação. Na transação POS o cupom utilizado deverá ser a via do estabelecimento.

Seguindo o exemplo do cupom de uma transação da CIELO da **Figura 8**, os campos serão mapeados da forma conforme abaixo.

ARQUIVO CONCILIAÇÃO					FUNÇÃO ScopeObtemCampoExt2			CUPOM
DESCRIÇÃO	**POS**	**TAM**	**TIPO**	**COMENTÁRIOS**	**MÁSCARA 1**	**MÁSCARA 2**	**MÁSCARA 3**	**CAMPO**
Tipo de registro	1	1	N	Constante ‘1’	NA*	NA*	NA*	NA*
Numero do PV	2	15	N	Código do Estabelecimento	0x00000800	NA*	NA*	NA*
Data da venda	17	8	D	Data da venda (YYYYMMDD)	NA*	0x00000008	NA*	Data
NSU	25	6	A	Num. seqüencial único	0x00000004	NA*	NA*	DOC
NSU do Host	31	15	A	NSU complementar	0x00004000	NA*	NA*	NA*
Código de autorização	41	14	A	Código de autorização	0x00000100	NA*	NA*	AUT
Valor da compra	51	15(2)	N	Valor da operação	0x00000002	NA*	NA*	VALOR
Número do cartão	66	22	A	Número do cartão	0x00000001	NA*	NA*	6 primeiros e 4 últimos dígitos do cartão
Status da transação	85	1	A	Transações OK, preencher com a letra “O”	NA*	NA*	NA*	NA*
Código Empresa	86	4	N	Formato “0000”	NA*	0x08000000	NA*	NA*

Código Filial	90	4	N	Formato “0000”	NA*	NA*	NA*	NA*
Qtd Parcelas	94	2	N	Quantidade de parcelas. Se venda for à vista, preencher ‘00’	0x00001000	NA*	NA*	Número de parcelas
Forma Captura	96	1	A	“P”-POS, “I”-Internet, “M”-Manual, “T”-TEF ou “O” Outros.	NA*	NA*	0x00000080	NA*
Código Bandeira	97	3	N	Código da Bandeira	0x00040000	NA*	NA*	Bandeira
Código Rede	100	3	N	Código da Rede	0x00400000	NA*	NA*	Rede
Código Serviço	103	5	N	Código do Serviço	0x00080000	NA*	NA*	Venda
RESERVADO	108	89	A	RESERVADO	NA*	NA*	NA*	NA*

* NA – não se aplica

Para ilustrar o mapeamento, considera-se o código da empresa e o código da filial do PDV sendo “0001” e “0001”, respectivamente, e o cupom de exemplo. Após a transação POS do SCOPE Client e a formatação da automação comercial, o cupom deverá resultar no registro com exibido na próxima tabela.

CUPOM	FUNÇÃO ScopeObtemCampoExt2				ARQUIVO
CAMPO	MÁSCARA 1	MÁSCARA 2	MÁSCARA 3	RETORNO	REGISTRO
-	-	-	-	-	"1"
-	0x00000800	0x00000000	0x00000000	" "	" "
041212	0x00000000	0x00000008	0x00000000	"041212"	"20121204"
096044	0x00000004	0x00000000	0x00000000	"096044"	"096044"
-	0x00004000	0x00000000	0x00000000	""	" "
095023	0x00000100	0x00000000	0x00000000	"095023"	"095023 "
1000	0x00000002	0x00000000	0x00000000	"1000"	"0000000000001000"
444054-2871	0x00000001	0x00000000	0x00000000	"444054-2871"	"444054-2871 "
-	-	-	-	-	"O"
-	0x00000000	0x08000000	0x00000000	"0001"	"0001"
-	-	-	-	-	"0001"
2	0x00001000	0x00000000	0x00000000	"2"	"02"
-	0x00000000	0x00000000	0x00000080	"P"	"P"
ELECTRON	0x00040000	0x00000000	0x00000000	"21"	"021"
CIELO	0x00400000	0x00000000	0x00000000	"102"	"102"
Débito parcelado	0x00080000	0x00000000	0x00000000	"134"	"134"
-	-	-	-	-	" ... "

Cupom REDECARD X ScopeObtemCampoExt2 X arquivo

Como no tópico anterior, nesse são mapeados os campos de cupons da rede REDECARD com as constantes da função `ScopeObtemCampoExt2` do SCOPE Client e os campos do registro do arquivo de conciliação. Na transação POS o cupom utilizado deverá ser a via do estabelecimento.

Figura 9: cupom da rede REDECARD

Seguindo o exemplo do cupom de uma transação da REDECARD da Figura 9, os campos serão mapeados da forma conforme tabela.

ARQUIVO CONCILIAÇÃO					FUNÇÃO ScopeObtemCampoExt2			CUPOM
DESCRIÇÃO	**POS**	**TAM**	**TIPO**	**COMENTÁRIOS**	**MÁSCARA 1**	**MÁSCARA 2**	**MÁSCARA 3**	**CAMPO**
Tipo de registro	1	1	N	Constante ‘1’	NA*	NA*	NA*	NA*
Numero do PV	2	15	N	Código do Estabelecimento	0x00000800	NA*	NA*	NA*
Data da venda	17	8	D	Data da venda (YYYYMMDD)	NA*	0x00000008	NA*	Data
NSU	25	6	A	Num. seqüencial único	0x00000004	NA*	NA*	NA*
NSU do Host	31	15	A	NSU complementar	0x00004000	NA*	NA*	CV
Código de autorização	41	14	A	Código de autorização	0x00000100	NA*	NA*	AUTO
Valor da compra	51	15(2)	N	Valor da operação	0x00000002	NA*	NA*	TOT. APROVADO
Número do cartão	66	22	A	Número do cartão	0x00000001	NA*	NA*	6 primeiros opcionais e 4 últimos obrigatórios dígitos do cartão
Status da transação	85	1	A	Transações OK, preencher com a letra “O”	NA*	NA*	NA*	NA*
Código Empresa	86	4	N	Formato “0000”	NA*	0x08000000	NA*	NA*
Código Filial	90	4	N	Formato “0000”	NA*	NA*	NA*	NA*
Qtd Parcelas	94	2	N	Quantidade de parcelas. Se venda for à vista, preencher ‘00’	0x00001000	NA*	NA*	Número de parcelas
Forma Captura	96	1	A	“P”-POS, “I”-Internet, “M”-Manual, “T”-TEF ou “O” Outros.	NA*	NA*	0x00000080	NA*

Código Bandeira	97	3	N	Código da Bandeira	0x00040000	NA*	NA*	Bandeira
Código Rede	100	3	N	Código da Rede	0x00400000	NA*	NA*	Rede
Código Serviço	103	5	N	Código do Serviço	0x00080000	NA*	NA*	Venda
RESERVADO	108	89	A	RESERVADO	NA*	NA*	NA*	NA*

* NA – não se aplica

Para ilustrar o mapeamento, considera-se o código da empresa e o código da filial do PDV sendo “0001” e “0001”, respectivamente, e o cupom de exemplo. Após a transação POS do SCOPE Client e a formatação da automação comercial, o cupom deverá resultar no registro com exibido nessa tabela.

CUPOM	FUNÇÃO ScopeObtemCampoExt2				ARQUIVO
CAMPO	MÁSCARA 1	MÁSCARA 2	MÁSCARA 3	RETORNO	REGISTRO
-	-	-	-	-	"1"
-	0x00000800	0x00000000	0x00000000	"000000000000000"	"000000000000000"
121212	0x00000000	0x00000008	0x00000000	"121212"	"20121212"
-	0x00000004	0x00000000	0x00000000	"096044"	"096044"
499931944	0x00004000	0x00000000	0x00000000	"499931944"	"499931944 "
018833	0x00000100	0x00000000	0x00000000	"018833"	"018833 "
1330	0x00000002	0x00000000	0x00000000	"1330"	"000000000001330"
8096	0x00000001	0x00000000	0x00000000	"000000-8096"	"000000-8096 "
-	-	-	-	-	"O"
-	0x00000000	0x08000000	0x00000000	"0001"	"0001"
-	-	-	-	-	"0001"
-	0x00001000	0x00000000	0x00000000	"1"	"01"
-	0x00000000	0x00000000	0x00000080	"P"	"P"
MAESTRO	0x00040000	0x00000000	0x00000000	"8"	"008"
REDECARD	0x00400000	0x00000000	0x00000000	"103"	"103"
Débito à vista	0x00080000	0x00000000	0x00000000	"6"	"006"
-	-	-	-	-	“0000...000”

Outros cupons X ScopeObtemCampoExt2 X arquivo

Pela configuração do scope.ini, o SCOPE Client poderá coletar dados de cupom de outras redes. No entanto, os nomes dos campos coletados serão genéricos.

ARQUIVO CONCILIAÇÃO					FUNÇÃO ScopeObtemCampoExt2			CUPOM	
DESCRIÇÃO	POS	TAM	TIPO	COMENTÁRIOS	MÁSCARA 1	MÁSCARA 2	MÁSCARA 3	CAMPO	OBRIG.
Tipo de registro	1	1	N	Constante ‘1’	NA*	NA*	NA*	NA*	NA*
Numero do PV	2	15	N	Código do Estabelecimento	0x00000800	NA*	NA*	NA*	NA*
Data da venda	17	8	D	Data da venda (YYYYMMDD)	NA*	0x00000008	NA*	Data	SIM
NSU	25	6	A	Num. seqüencial único	0x00000004	NA*	NA*	NA*	NÃO
NSU do Host	31	15	A	NSU complementar	0x00004000	NA*	NA*	CV	NÃO
Código de autorização	41	14	A	Código de autorização	0x00000100	NA*	NA*	AUTO	NÃO
Valor da compra	51	15(2)	N	Valor da operação	0x00000002	NA*	NA*	TOT. APROVADO	SIM
Número do cartão	66	22	A	Número do cartão	0x00000001	NA*	NA*	6 primeiros dígitos do cartão	NÃO

								4 últimos dígitos do cartão	SIM
Status da transação	85	1	A	Transações OK, preencher com a letra “O”	NA*	NA*	NA*	NA*	NA*
Código Empresa	86	4	N	Formato “0000”	NA*	0x08000000	NA*	NA*	NA*
Código Filial	90	4	N	Formato “0000”	NA*	NA*	NA*	NA*	NA*
Qtd Parcelas	94	2	N	Quantidade de parcelas. Se venda for à vista, preencher ‘00’	0x00001000	NA*	NA*	Número de parcelas	SIM
Forma Captura	96	1	A	“P”-POS, “I”-Internet, “M”-Manual, “T”-TEF ou “O” Outros.	NA*	NA*	0x00000080	NA*	NA*
Código Bandeira	97	3	N	Código da Bandeira	0x00040000	NA*	NA*	Bandeira	SIM
Código Rede	100	3	N	Código da Rede	0x00400000	NA*	NA*	Rede	SIM
Código Serviço	103	5	N	Código do Serviço	0x00080000	NA*	NA*	Venda	SIM
RESERVADO	108	89	A	RESERVADO	NA*	NA*	NA*	NA*	NA*

Formatação para o arquivo de conciliação

Observa-se nos exemplos a automação comercial precisará tratar alguns campos, formatando-os para gravar no arquivo de conciliação. Será necessária a formatação nos campos:

- Tipo de registro: preencherá sempre com o dígito ‘1’.
- Data da transação: o SCOPE Cliente fornece a data no formato DDMMAA e a automação alterará para o formato AAAAMMDD.
- NSU do host: completará com espaços à direita.
- Código de autorização: completará com espaços à direita.
- Valor da transação: completará com zeros à esquerda.
- Número do cartão: completará com espaços à direita.
- Status da transação: preencherá sempre com a letra “O”.
- Código da filial: preencherá com o código da filial usado na função ScopeOpen.
- Número de parcelas: completará com zeros à esquerda.
- Código da rede: completará com zeros à esquerda.
- Código da bandeira: completará com zeros à esquerda.
- Reservado: preencherá sempre com espaços à direita.

Funções diversas

Neste capítulo, abordamos diversas funções de uso frequente por diversas empresas, mas que não conseguimos categorizá-las.

Dados da transação

Após uma transação ou até mesmo durante o processamento desta, pode-se querer guardar dados da transação junto aos dados de venda, utilizados pela aplicação para controle ou relatório das vendas.

No caso da interface HLAPI, é muito útil obter esses dados, já que é o SCOPE quem coleta e ela não conhece nenhum dado. Então ao final da transação, a aplicação poderá obter do SCOPE alguns dados que estão disponíveis.

Já na interface coleta, quase todos os dados da transação a aplicação consegue obter, uma vez que é ela que provê a entrada de dados e repassa o que foi coletado para o SCOPE. A aplicação pode conhecer o dado coletado a partir do código do estado de coleta que o SCOPE passa para ela coletar.

Para obter estes dados, primeiramente, a aplicação solicita um *handle* para a transação. Uma vez obtido este *handle*, a aplicação solicita os dados da transação, um de cada vez ou todos de uma única vez.

Obtendo *handle*

A aplicação deverá solicitar o *handle* com a função ScopeObtemHandle(). Dependendo do momento em que coletará os dados, o parâmetro desta função será diferente:

- valor **0**: este valor deverá ser passado para uma referência da última transação ou da transação em andamento após a solicitação de autorização;
- valor **8**: este valor é utilizado para a obtenção de dados de uma transação após queda de energia.
- valor **9**: este valor é usado quando o momento de obtenção de algum dado acontecer durante o processamento da transação, antes da solicitação de autorização.

Protótipo

```
LONG EXPORT ScopeObtemHandle (LONG Desloc)
```

Parâmetros

[in]	LONG	Desloc	Momento de obtenção do handle

Retorno

Valores maiores que 0xFFFF significam valores válidos de *handle*. Qualquer outro valor significa um código de erro (Ver tabela de código de retorno).

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	

0xFA01	64001	Parâmetro 1 inválido
0xFB01	64257	Não foi possível criar a "*thread*" na coleta de dados
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE05	65029	API ainda não fez nenhuma transação após a inicialização. A aplicação está tentando obter um *handle* sem ter feito nenhuma transação desde última conexão com o ScopeSRV
0xFE0D	65037	Não há arquivo com dados da transação anterior salvo

Exemplo

```
...
LONG handle;
// processa a transação
handle = ScopeObtemHandle(0);
if(handle > 0xFFFF)
{
   // obtém os dados desejados
}
else
{
   // erro! não conseguiu o handle
}
...
// fecha a sessão
...
```

Obtendo os campos

Após obter um *handle* válido, a aplicação está apta para o recebimento dos dados da transação. Estes dados podem ser obtidos de uma vez só ou campo a campo, ou até mesmo agrupados em partes. Isto é conseguido através de campos com combinação de bits.

Cada bit das variáveis de máscara passadas para o SCOPE possui um significado que podem ser combinados de qualquer maneira. A relação completa de campos e os bits que representam estes campos podem ser encontrados na seção Dados disponíveis das transações. Para casos em que a aplicação solicitar vários dados, o SCOPE os retornará num único buffer separado por um caractere separador que a aplicação passará para o SCOPE usar. Mesmo que a aplicação solicitar um dado que não exista na transação, o SCOPE separará o campo em que estaria o dado com o caractere separador.

A sequência dos dados que serão devolvidos para a aplicação será a partir da máscara 1, depois para a máscara 2 e logo depois para a máscara 3 na sequência do bit menos significativo para o mais significativo. É importante observar que a máscara 3 somente pode ser obtida através da função ScopeObtemCampoExt2(), juntamente com as máscaras 1 e 2

Protótipo

```
LONG EXPORT ScopeObtemCampoExt (LONG Handle,
                                LONG Masc1,
                                LONG Masc2,
```

```
                                    char FieldSeparator,
                                    char *Buffer)
```

Parâmetros

[in]	LONG	Handle	Handle da transação, uma referência ao registro contendo os dados da transação, que deve ser um valor retornado pela função ScopeObtemHandle()
[in]	LONG	Masc1	Máscara de bits indicando os campos de dados da transação a serem recuperados que pode ser um valor tipo bitwise (Ver bits relacionados na tabela Dados disponíveis das transações na parte da máscara 1)
[in]	LONG	Masc2	Máscara de bits indicando os campos de dados da transação a serem recuperados que pode ser um valor tipo bitwise (Ver bits relacionados na tabela Dados disponíveis das transações na parte da máscara 2)
[in]	char	FieldSeparator	Caractere de separação usado quando Masc1 e/ou Masc2 forem do tipo bitwise recuperando mais de um campo no mesmo buffer
[out]	string	Buffer	Buffer de destino que conterá os dados que o SCOPE retornará

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFA03	64003	Parâmetro 3 inválido
0xFA04	64004	Parâmetro 4 inválido
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFE01	65025	SCOPE API não foi inicializada corretamente

Exemplo

```
...
long h;
char aux[128];
handle = ScopeObtemHandle(0);
if(handle > 0xFFFF)
{
   // obtendo o nsu da transação
   memset(aux, '\0', sizeof(aux));
   ScopeObtemCampoExt(h, 0x00000004, 0x00 , ';', aux);
   ArmazenaCampo(aux);  // exemplo de retorno "015236:"

   // obtendo o código da operadora de celular
```

```
   memset(aux, '\0', sizeof(aux));
   ScopeObtemCampoExt(h, 0x00, 0x00004000, ':', aux);
   ArmazenaCampo(aux);  // exemplo de retorno "02:"

   // código e nome da bandeira
   memset(aux, '\0', sizeof(aux));
   ScopeObtemCampoExt(h, 0x00040000 | 0x00800000, 0x00 , ':', aux);
   ArmazenaCampo(aux);  // exemplo de retorno "002:Mastercard:"
}
else
{
   // erro! não conseguiu o handle
}
...
```

Protótipo

```
LONG EXPORT ScopeObtemCampoExt2 (LONG Handle,
                                 LONG Masc1,
                                 LONG Masc2,
                                 LONG Masc3,
                                 char FieldSeparator,
                                 char *Buffer)
```

Parâmetros

[in]	LONG	Handle	Handle da transação, uma referência ao registro contendo os dados da transação, que deve ser um valor retornado pela função ScopeObtemHandle()
[in]	LONG	Masc1	Máscara de bits indicando os campos de dados da transação a serem recuperados que pode ser um valor tipo bitwise (Ver bits relacionados na tabela Dados disponíveis das transações na parte da máscara 1)
[in]	LONG	Masc2	Máscara de bits indicando os campos de dados da transação a serem recuperados que pode ser um valor tipo bitwise (Ver bits relacionados na tabela Dados disponíveis das transações na parte da máscara 2)
[in]	LONG	Masc3	Máscara de bits indicando os campos de dados da transação a serem recuperados que pode ser um valor tipo bitwise (Ver bits relacionados na tabela Dados disponíveis das transações na parte da máscara 3)
[in]	char	FieldSeparator	Caractere de separação usado quando Masc1 e/ou Masc2 forem do tipo bitwise recuperando mais de um campo no mesmo buffer
[out]	string	Buffer	Buffer de destino que conterá os dados que o SCOPE retornará

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFA03	64003	Parâmetro 3 inválido
0xFA04	64004	Parâmetro 4 inválido
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFE01	65025	SCOPE API não foi inicializada corretamente

Exemplo

```
...
long h;
char aux[128];
handle = ScopeObtemHandle(0);
if(handle > 0xFFFF)
{
   // obtendo o nsu da transação
   memset(aux, '\0', sizeof(aux));
   ScopeObtemCampoExt2(h, 0x00000004, 0x00 , 0x00 ,';', aux);
   ArmazenaCampo(aux);  // exemplo de retorno "015236:"

   // obtendo o código da operadora de celular
   memset(aux, '\0', sizeof(aux));
   ScopeObtemCampoExt2(h, 0x00, 0x00004000, 0x00 ':', aux);
   ArmazenaCampo(aux);  // exemplo de retorno "02:"

   // código e nome da bandeira e dados da plataforma promocional
   memset(aux, '\0', sizeof(aux));
   ScopeObtemCampoExt2(h, 0x00040000 | 0x00800000, 0x00 , 0x00000001 ':',
aux);
   ArmazenaCampo(aux);  // exemplo de retorno "002:Mastercard:"
}
else
{
   // erro! não conseguiu o handle
}
...
```

CUIDADO: tente alocar um buffer razoavelmente grande conforme os dados que se deseja recuperar, pois o SCOPE não alocará buffer.

Fornecendo informações extras para a transação

Para passar alguma informação que não é padrão do fluxo da transação, existe a função ScopeForneceCampo() como meio de passar esses dados. São várias as informações que são recebidas por esta função.

Descrição dos tipos de dados

O SCOPE conhece cada tipo de dado pelo valor recebido no primeiro parâmetro, pois cada tipo possui um significado diferente, não tendo relação entre si. Os dados são esperados no segundo parâmetro (buffer).

No arquivo *ScopeApi.h* que segue com as bibliotecas do SCOPE Client, existe uma enumeração contendo todos os valores que indicam cada tipo de dado, cuja declaração é:

```
typedef enum {
     SCOPE_DADO_MIN,
     SCOPE_DADO_EMPRESA,
     SCOPE_DADO_PLANO_LOJA,
     SCOPE_DADO_ATRIBUTOS_APLIC,
     SCOPE_DADO_TRILHA_01,
     SCOPE_DADO_REG_FORMA_PAGTO,
     SCOPE_DADO_AUT_SUPERVISOR,
     SCOPE_DADO_IMPRIME_CHEQUE,
     SCOPE_DADO_SEPARADOR_LINHA,
     SCOPE_DADOS_APLIC,
     SCOPE_DADOS_PAGAMENTO,
     SCOPE_DADOS_APLIC_CBD,
     SCOPE_DADO_DATA_MOVIMENTO,
     SCOPE_DADO_NSU_CUPOM_FISCAL,
     SCOPE_DADOS_AUTOMACAO,
     SCOPE_DADOS_COD_AUT_MEDICAMENTOS,
     SCOPE_DADOS_LISTA_MEDICAMENTOS,
     SCOPE_AUTOMACAO_PERMITE_SALDO_VOUCHER,
     SCOPE_COD_TABELA_PARCELE_MAIS,
     SCOPE_DADOS_PAGAMENTO_EX,
     SCOPE_AUTOMACAO_PARTICIPA_PPONLINE,
     SCOPE_PERMITIR_SAQUE,
     SCOPE_RESULTADO_VALIDACAO,
     SCOPE_DADOS_ECF_BENEFICIO,
     SCOPE_NOME_PRODUTO,
     SCOPE_DADOS_CARTAO_PRESENTE,
     SCOPE_DADOS_LISTA_PRECOS,
     SCOPE_DADOS_LISTA_MERCADORIAS,
     SCOPE_DADO_MAX,
```

```
} tDadoForneceCampo;
```

Onde:

- SCOPE_DADO_MIN (ID 0): limite inferior do domínio de valores que o primeiro parâmetro pode receber
- SCOPE_DADO_EMPRESA (ID 1): buffer onde os 7 primeiros bytes representam uma string com o número do cupom/nota fiscal e outros 7 bytes com a série da nota fiscal (exemplo: “123456’\0’598654’\0’” = {‘1’, ‘2’, ‘3’, ‘4’, ‘5’, ‘6’, ‘\0’, ‘5’, ‘9’, ‘8’, ‘6’, ‘5’, ‘4’, ‘\0’})
- SCOPE_DADO_PLANO_LOJA (ID 2): string com o máximo 12 bytes de comprimento contendo o valor total do ticket, dado complementar utilizado pela função ScopeCompraCDC (exemplo: “10000” para R$100,00)
- SCOPE_DADO_ATRIBUTOS_APLIC (ID 3): ver Atributos da aplicação
- SCOPE_DADO_TRILHA_01 (ID 4): string com a trilha 01 do cartão lida no PIN-Pad
- SCOPE_DADO_REG_FORMA_PAGTO (ID 5): string de no máximo 30 bytes de comprimento com a descrição do registrador de forma de pagamento
- SCOPE_DADO_AUT_SUPERVISOR (ID 6): string de no máximo 15 bytes de comprimento com a autorização do supervisor
- SCOPE_DADO_IMPRIME_CHEQUE (ID 7): string com 1 byte de comprimento informando se imprime (“S”) ou não (“N”) o cheque
- SCOPE_DADO_SEPARADOR_LINHA (ID 8): string com 1 byte de comprimento parametrizando o separador de linha do cupom (exemplo: “@”). Atualmente, suporta apenas o caractere '@', sendo que para qualquer outro caractere parametrizado, ou caso esta parametrização não seja efetuada, será considerado o caractere '\n' como finalizador de linha do cupom.
- SCOPE_DADOS_APLIC (ID 9): tipo de terminal (‘A’ para ATM ou ‘P’ para PDV) e usuário
- SCOPE_DADOS_PAGAMENTO (ID 10): dados do pagamento (confronte com SCOPE_DADOS_PAGAMENTO_EX)
- SCOPE_DADOS_APLIC_CBD (ID 11): desenvolvido para a rede CBD e utilizado também pela rede VR (SmartNet), sendo um string com o primeiro char informando o tipo de terminal (numero de ‘1’ a ’5’) e o restante da string representando a data de movimento no formato MMDD, como demonstrado a seguir:

Campo	Formato	Descrição
Tipo Terminal	n1	Tipos de Terminal que podem ser configurados: 1 = Frente de loja, (default) 2 = Magazine, 3 = Batimento/retaguarda 4 = Posto de gasolina 5 = WEB
Data de Movimento	n4	Data de Movimento no formato MMDD

- SCOPE_DADO_DATA_MOVIMENTO (ID 12): não mais suportado
- SCOPE_DADO_NSU_CUPOM_FISCAL (ID 13): não mais suportado

- SCOPE_DADOS_AUTOMACAO (ID 14): buffer onde os 21 primeiros bytes representam uma string com o fabricante do software de automação e outros 21 bytes com a versão do software da automação
- SCOPE_DADOS_COD_AUT_MEDICAMENTOS (ID 15): string com 13 bytes que contém o código de autorização da PBM
- SCOPE_DADOS_LISTA_MEDICAMENTOS (ID 16): buffer com a lista de medicamentos da PBM
- SCOPE_AUTOMACAO_PERMITE_SALDO_VOUCHER (ID 17): Através deste campo, a automação informa ao Scope se permite que o pagamento efetuado através de voucher utilize o valor disponível no saldo (neste caso deve informar 'S') ou aceita somente pagamento para os casos em que o saldo do voucher seja igual ou superior ao valor da transação (neste caso deve informar 'N'). Observe que este campo deve ser informado somente na abertura de conexão entre a aplicação e o Scope Cliente (ScopeOpen).
- SCOPE_COD_TABELA_PARCELE_MAIS (ID 18): Automação pode fornecer um código de tabela a ser usado na transação de Parcele Mais, indicando o custo financeiro.
- SCOPE_DADOS_PAGAMENTO_EX (ID 19): dados do pagamento para transações que necessitam do número do cartão (confronte com SCOPE_DADOS_PAGAMENTO)
- SCOPE_AUTOMACAO_PARTICIPA_PPONLINE (ID 20): responsável por informar se a automação comercial está apta a trabalhar com a plataforma promocional.
- SCOPE_PERMITIR_SAQUE (ID 21): em transações de débito à vista com a rede Cielo, ao fornecer o valor '1' indicará que o SCOPE pode pedir a digitação do valor do saque se configurado pela rede e ao fornecer '0' o SCOPE não perguntará pelo saque, mesmo configurado pela rede Cielo e no SCOPECNF.
- SCOPE_RESULTADO_VALIDACAO(ID 22): após a validação do saque pela automação, está passará o valor '1' aprovando o valor do saque digitado pelo portador do cartão no pinpad ou passará o valor '0' recusando o valor do saque solicitado.
- SCOPE_DADOS_ECF_BENEFICIO (ID 23): fornecer os dados da ECF e do comprovante para a transação.
- SCOPE_DADOS_LISTA_PRECOS (ID 26): fornecer os dados da Lista de Atualização de Preços das Mercadorias.
- SCOPE_DADOS_LISTA_MERCADORIAS (ID 27): fornecer os dados da Lista de Mercadorias Consumidas na Aplicação.
- SCOPE_DADO_MAX (ID 29): limite superior do domínio de valores que o primeiro parâmetro pode receber

Atributos da aplicação

Permite implementar atributos (campos) dinamicamente, que são vinculados as transações efetuadas pelo SCOPE. Existem três módulos envolvidos por esta funcionalidade:

- ScopeCNF: cadastrando os atributos (ver no documento de instalação, configuração e administração, informações sobre cadastramento)

- Aplicação do PDV: deverá usar a função ScopeForneceCampo()
- ScopeADM: irá exibir as transações vinculadas aos atributos cadastrados (ver o documento de instalação, configuração e administração para informações sobre relatório)

Exemplo: você poderá consultar as transações efetuadas pelo SCOPE, vinculadas a outras informações úteis a empresa, como: número da nota fiscal, do pedido, da compra, nome e telefone do cliente, etc.

O formato dos dados deve ser uma string que obedece ao seguinte padrão:

"nnc1tt1xx..x1c2tt2xx..x2cnttnxx..xn"

, onde:

- nn = quantidade de atributos (tamanho 2)
- c1 = código do atributo 1 (tamanho 2)
- tt1 = tamanho do atributo 1 (tamanho 3)
- xx..x1 = atributo 1 (tamanho tt1)
- c2 = código do atributo 2 (tamanho 2)
- tt2 = tamanho do atributo 2 (tamanho 3)
- xx..x2 = atributo 2 (tamanho tt2)
- ...
- cn = código do atributo n (tamanho 2)
- ttn = tamanho do atributo n (tamanho 3)
- xx..xn = atributo n (tamanho ttn)

IMPORTANTE: por razões de otimização de tráfego, o SCOPE Client envia estes dados para o servidor na solicitação da segunda pré-TEF. Portanto, a aplicação deve fornecer este buffer antes da realização da mesma.

IMPORTANTE: na interface HLAPI, a aplicação deve mandar suspender em um dos estados de coleta que antecedem a segunda pré-TEF.

Atrelando dados à transação

Passar algum dado ao SCOPE com a função ScopeForneceCampo() exige que ela seja chamada antes do envio da segunda pré-TEF.

Protótipo

```
LONG EXPORT ScopeForneceCampo (char TypeField, void *StructField)
```

Parâmetros

[in]	char	TypeField	Tipo de dado
[in]	void *	StructField	Ponteiro para os dados

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFB01	64257	Não foi possível criar a "*thread*" na coleta de dados
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE05	65029	API ainda não fez nenhuma transação após a inicialização. A aplicação está tentando obter um *handle* sem ter feito nenhuma transação desde última conexão com o ScopeSRV
0xFE0D	65037	Não há arquivo com dados da transação anterior salvo
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF

Exemplo

Digamos que a cada venda paga com o cartão de crédito, a empresa deseja que seja atrelada a transação o número de telefone do cliente e o número do pedido. Para isso, o administrador do SCOPE criou dois atributos no ScopeCNF (figura 7).

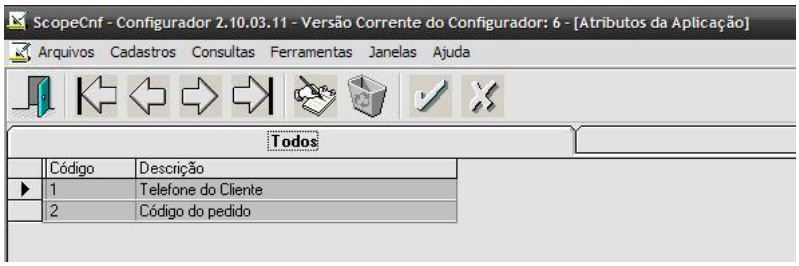


Figura 10: atributos da aplicação cadastrados no ScopeCNF

A aplicação coletará estes dados sempre que houver uma venda com cartão de crédito. Exemplificando na linguagem C uma compra com cartão de crédito (interface Alto Nível - HLAPI):

```
...
LONG RC;
char szAtributosAplic[40];
...
// operador selecionou crédito
...
// abre sessão de TEF
...
ScopeSuspend(EC_COLETA_CARTAO);
ScopeCompraCartaoCredito("100", "0");
...
// processa a transação
if (RC == RCS_EXISTE_TRN_SUSPENSA)
{
   // coletou o telefone do cliente e número do pedido
   ...
   // montou o buffer do atributo da aplicação
   // (ex. "0201014(11) 6097-123402006123456" e guardou em szAtributosAplic
   ...
   ScopeForneceCampo(SCOPE_DADO_ATRIBUTOS_APLIC, szAtributosAplic);
   ScopeResume();
```

```
}
...
// encerra sessão de TEF
```

Separando o buffer do exemplo acima conforme o padrão descrito, obtemos:

- 02 – 2 campos no buffer
- 01 – este campo representa o que está cadastrado como código 1 no ScopeCNF
- 014 – o atributo que seguirá contém 14 bytes
- (11) 6097-1234 – valor do atributo de código 1 com 14 bytes
- 02 – este campo representa o que está cadastrado como código 2 no ScopeCNF
- 006 – o atributo que seguirá contém 6 bytes
- 123456 – valor do atributo de código 2 com 6 bytes

ScopeGetLastMsg

Obtém as últimas mensagens a serem mostradas para o operador e/ou cliente.

Protótipo

```
LONG EXPORT ScopeGetLastMsg (ptCOLETA_MSG ptParamColetaMsg)
```

Parâmetros

[out]	Ponteiro para uma área com o formato da estrutura stCOLETA_MSG	ptParamColetaMsg	Estrutura contendo as mensagens do operador e do cliente (tanto a linha 1 como a linha 2). As mensagens são finalizadas pelo caractere nulo (‘\0’)

Estruturas de apoio

```
typedef struct _stCOLETA_MSG {
   char Op1[64];
   char Op2[64];
   char Cl1[64];
   char Cl2[64];
} stCOLETA_MSG, *ptCOLETA_MSG;
```

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFB01	64257	Não foi possível criar a “*thread*” na coleta de dados
0xFE01	65025	SCOPE API não foi inicializada corretamente

ScopeGetCheque

Obtém os parâmetros para impressão de cheque.

Protótipo

```
LONG EXPORT ScopeGetCheque(ptPARAM_CHEQUE ptParamCheque)
```

Parâmetros

[out]	Ponteiro para uma área com o formato da estrutura ptPARAM_CHEQUE	ptParamCheque	Dados para impressão do cheque

Estruturas de apoio

```
typedef struct  _stPARAM_CHEQUE {
   char Banco[4];        // número do banco
   char Agencia[5];      // número da agência
   char NumCheque[13];   // número do cheque
   char Valor[13];       // valor do cheque
   char BomPara[9];      // data do cheque
   char CodAut[11];      // cód. de autor. retornado pelas autorizadoras
   char Municipio[41];   // município
   short Ordem;          // reservado
} stPARAM_CHEQUE, *ptPARAM_CHEQUE;
```

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE04	65028	Não existe transação suspensa

ScopeAtualizaValor

Atualiza ou fornece o novo valor da transação. Esta função deve ser executada após a função de transação e antes do envio da solicitação de autorização para a Autorizadora.

Protótipo

```
LONG EXPORT ScopeAtualizaValor(char *Valor)
```

Parâmetros

[in]	String	Valor	Valor da transação. Deve ser uma string de até 12 bytes.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFE01	65025	SCOPE API não foi inicializada corretamente

ScopeGarantiaDescontoCheque

Aciona o SCOPE client para efetuar uma transação de Garantia de Cheque ou de Desconto Antecipado de Cheque (Factoring).

Protótipo

```
LONG EXPORT ScopeGarantiaDescontoCheque (char *Valor)
```

Parâmetros

[in]	String	Valor	Valor da transação. Deve ser uma string de até 12 bytes.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE05	65029	API ainda não fez nenhuma transação após a inicialização. A aplicação está tentando obter um *handle* sem ter feito nenhuma transação desde última conexão com o ScopeSRV
0xFE0D	65037	Não há arquivo com dados da transação anterior salvo
0xFF36	65334	Erro no envio da confirmação ou desfazimento da transação anterior
0xFF3A	65338	Erro interno na execução da coleta

ScopeTransacaoFinanceira

Aciona o SCOPE Client para efetuar uma transação financeira.

Protótipo

```
LONG EXPORT ScopeTransacaoFinanceira (char *Valor, WORD Servico)
```

Parâmetros

[in]	String	Valor	Valor da transação
[in]	WORD	Serviço	Informa qual o serviço a ser adotado. Caso 0 (zero) a coleta do serviço será feita pelo procedimento usual (ver Códigos dos serviços)

Retorno
Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64002	Parâmetro 2 inválido
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFB01	64257	Não foi possível criar a "*thread*" na coleta de dados
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual

ScopeInvestimento

Aciona o SCOPE Client para efetuar um investimento.

Protótipo

```
LONG EXPORT ScopeInvestimento (char *Valor, WORD Servico)
```

Parâmetros

[in]	String	Valor	Valor da transação
[in]	WORD	Servico	Informa qual o serviço a ser adotado. Caso 0 (zero) a coleta do serviço será feita pelo procedimento usual (ver Códigos dos Serviços)

Retorno
Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64002	Parâmetro 2 inválido
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE05	65029	API ainda não fez nenhuma transação após a inicialização. A aplicação está tentando obter um *handle* sem ter feito nenhuma transação desde última conexão com o ScopeSRV
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).

0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFF3A	65338	Erro interno na execução da coleta
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFF5E	65374	Erro ao desmontar a estrutura ISO
0xFFFF	65535	Erro genérico

ScopeObtemCartaoInvestimento

Aciona o SCOPE Client para obter o número do cartão a partir do CPF do cliente. Viabiliza efetuar uma transação de aplicação pelo número do CPF.

Protótipo

```
LONG EXPORT ScopeObtemCartaoInvestimento (char *CPF,
                                         char *Buf,
                                         WORD TamBuf)
```

Parâmetros

[in]	String	CPF	CPF do cliente
[out]	String	Buf	Área reservada pela aplicação para receber o cartão (ou lista de cartões). O layout dos dados foi definido pelo SAB (o SCOPE não critica nem altera este layout).
[in]	WORD	TamBuf	A aplicação deve informar ao SCOPE o tamanho da área reservada

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	**Decimal**	
0xFA01	64001	Parâmetro 1 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line

0xFFFF	65535	Erro genérico
0xFF5E	65374	Erro ao desmontar a estrutura ISO

ScopeResumoOperacoes

Utilizada nos casos em que a rede ou correspondente bancário oferece a possibilidade de obtenção de um resumo das operações realizadas.

Protótipo

```
LONG EXPORT ScopeResumoOperacoes (WORD CodServico, WORD CodBandeira)
```

Parâmetros

[in]	WORD	CodServico	Código do serviço do produto a ser executado. Use 0 (zero) para indicar que o código do serviço deverá ser coletado durante o fluxo da transação.
[in]	WORD	CodBandeira	Código da bandeira do produto a ser executado. Use 0 (zero) para indicar que o código da bandeira deverá ser coletado.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE05	65029	API ainda não fez nenhuma transação após a inicialização. A aplicação está tentando obter um *handle* sem ter feito nenhuma transação desde última conexão com o ScopeSRV
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFE0D	65037	Não há arquivo com dados da transação anterior salvo
0xFB01	64257	Não foi possível criar a "*thread*" na coleta de dados
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
*0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual

0xFF36	65334	Erro no envio da confirmação ou desfazimento da transação anterior
0xFF3A	65338	Erro interno na execução da coleta
0xFF5E	65374	Erro ao desmontar a estrutura ISO
0xFF60	65376	Função indisponível
0xFFFF	65535	Erro genérico

ScopePagamento

Utilizada para pagamento de conta (título ou convênio) ou fatura, através de uma rede autorizadora ou por intermédio de correspondente bancário. Os serviços atuais disponíveis são “pagamento de conta com cartão”, “pagamento de conta sem cartão” e “pagamento de fatura”. O pagamento de conta com cartão é usado quando a rede exige uma determinada bandeira (Electron, Cheque Eletrônico), sendo que neste caso o SCOPE valide o BIN do cartão. Pagamento de conta sem cartão é quando o pagamento pode ser realizado através de dinheiro, cheque ou TEF externa (outra transação realizada separadamente). Pagamento de fatura é usado para o “recebimento próprio” de fatura de cartão de crédito. Neste caso, o código de barras não é obrigatório, já que o pagamento pode ser através do número do cartão (digitado ou lido).

Protótipo

```
LONG EXPORT ScopePagamento (WORD CodServico, WORD CodBandeira)
```

Parâmetros

[in]	WORD	CodServico	Código do serviço do grupo “Pagamento de Contas” a ser executado. Use 0 (zero) para indicar que a decisão do serviço será através de coleta.
[in]	WORD	CodBandeira	Código da bandeira do produto a ser executado. Use 0 (zero) para indicar que o código da bandeira deverá ser coletado.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE05	65029	API ainda não fez nenhuma transação após a inicialização. A aplicação está tentando obter um *handle* sem ter feito nenhuma transação desde última conexão com o ScopeSRV
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado

0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFE0D	65037	Não há arquivo com dados da transação anterior salvo
0xFB01	64257	Não foi possível criar a "*thread*" na coleta de dados
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
*0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFF36	65334	Erro no envio da confirmação ou desfazimento da transação anterior
0xFF3A	65338	Erro interno na execução da coleta
0xFF5E	65374	Erro ao desmontar a estrutura ISO
0xFF60	65376	Função indisponível
0xFFFF	65535	Erro genérico

ScopeServicoTecnico

Utilizada para execução de um serviço técnico disponível para determinada rede autorizadora. Temos os seguintes serviços técnicos: Baixa de OS, Teste de Comunicação, Estatistica e Injeção de Chaves DUKPT.

O serviço técnico de Injeção de Chaves DUKPT insere a senha DUKPT no pinpad através de uma transação 0900/0800 Injeção de Chaves e sua resposta 0910/0810 quando o pinpad não possui a senha, isto é o serviço não atualiza senha; para isso utiliza-se a função VerificaChaveDUKPT para verificar se o pinpad possue senha DUKPT.

Protótipo

```
LONG EXPORT ScopeServicoTecnico (WORD CodServico, WORD CodBandeira)
```

Parâmetros

[in]	WORD	CodServico	Código do serviço do grupo "Técnico" a ser executado. Use 0 (zero) para indicar que a decisão do serviço será através de coleta.
[in]	WORD	CodBandeira	Código da bandeira do produto a ser executado. Use 0 (zero) para indicar que o código da bandeira deverá ser coletado.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido

0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE05	65029	API ainda não fez nenhuma transação após a inicialização. A aplicação está tentando obter um *handle* sem ter feito nenhuma transação desde última conexão com o ScopeSRV
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFE0D	65037	Não há arquivo com dados da transação anterior salvo
0xFB01	64257	Não foi possível criar a "*thread*" na coleta de dados
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
*0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFF36	65334	Erro no envio da confirmação ou desfazimento da transação anterior
0xFF3A	65338	Erro interno na execução da coleta
0xFF5E	65374	Erro ao desmontar a estrutura ISO
0xFF60	65376	Função indisponível
0xFFFF	65535	Erro genérico

ScopeAtualizaParametrosChip

Utilizada quando parâmetros do chip do cartão necessitam ser atualizados. Esta função inicia o fluxo de coleta necessário para a atualização de parâmetros do chip do cartão.
Redes que utilizam esta função são: Ticket Edenred e Cielo R2014.

Coleta básica da transação

- Solicita ao portador a inserção do cartão no pinpad
- Solicita a senha
- Solicita a retirada do cartão do pinpad

Protótipo

```
LONG EXPORT ScopeAtualizaParametrosChip(char *UsoFuturo1,
                                        char *UsoFuturo2)
```

Parâmetros

[in]	char	UsoFuturo1	Reservado para uso futuro. Usar como parâmetro uma string vazia.
[in]	char	UsoFuturo2	Reservado para uso futuro. Usar como parâmetro uma string vazia.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	**Decimal**	
0xFB01	64257	Não foi possível criar a "*thread*" na coleta de dados
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE05	65029	API ainda não fez nenhuma transação após a inicialização. A aplicação está tentando obter um *handle* sem ter feito nenhuma transação desde última conexão com o ScopeSRV
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
*0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFF36	65334	Erro no envio da confirmação ou desfazimento da transação anterior
0xFFFF	65535	Erro genérico

ScopeVersao

Esta função retorna a versão do Scope.

Protótipo

```
LONG EXPORT ScopeVersao(char *_VersaoScope, int_TamBufVersao)
```

Parâmetros

[out]	char	_VersaoScope	Informa o valor da versão do Scope em formato *string* e *null-terminated*. Em caso de erro, o *buffer* passado não sofre alteração.
[in]	int	_TamBufVersao	Contém o valor do tamanho da área que contém a versão. Deve conter o valor mínimo de 12 bytes.

Retorno

Ver tabela de código de retorno.

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido

ScopeAtualizaPrecosMercadorias

Utilizada quando os Preços das Mercadorias do Estabelecimento necessitam ser atualizados na rede Ticket Edenred. Esta função inicia o fluxo de coleta necessário para a Atualização de Preços das Mercadorias.

Redes que utilizam esta função são: Ticket Edenred.

Protótipo

```
LONG EXPORT ScopeAtualizaPrecosMercadorias ( WORD _CodBandeira,
                                             char *UsoFuturo1 )
```

Parâmetros

[in]	WORD	_CodBandeira	Código da Bandeira (Ex. "228" - Ticket Car)
[in]	char	UsoFuturo1	Reservado para uso futuro. Usar como parâmetro uma string vazia.

Retorno

Ver tabela de código de retorno.

Estados de coleta

A tabela abaixo mostra os estados de coleta que o fluxo poderá retornar, no modo coleta

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFCCA	64714	Coleta da Lista de Atualização de Preços de Mercadorias

Possíveis Retornos de Erros

Códigos Retorno		Significado
Hexadecimal	**Decimal**	
0xFB01	64257	Não foi possível criar a "*thread*" na coleta de dados
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE05	65029	API ainda não fez nenhuma transação após a inicialização. A aplicação está tentando obter um *handle* sem ter feito nenhuma transação desde última conexão com o ScopeSRV
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF0A	65290	Banco de dados off-line
0xFF0C	65292	Transação não implementada
*0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFF36	65334	Erro no envio da confirmação ou desfazimento da transação anterior

Redes com tratamentos específicos

O objetivo de capítulo é abordar peculiaridades de tratamentos para algumas redes autorizadoras como a coleta de dados no fluxo do SCOPE ou o relacionamento dos produtos da rede e as transações do SCOPE.

Banrisul EMV

Esta é a implementação da nova especificação do Banrisul e que trata principalmente a utilização de cartões com chip. Com esta implementação, será possível realizar as seguintes transações do Banrisul no SCOPE:

- Débito à vista, parcelado e pré-datado com o cartão Banricompras;
- Débito à vista e parcelado com o cartão Banco SIM;
- Crédito 1 minuto, que é uma transação de CDC;
- Débito voucher de convênio Refeisul alimentação e refeição;
- Consulta saldo de convênio Refeisul alimentação e refeição;
- Débito voucher de convênio Refeisul combustível
- Crédito com cartão convênio Private Label

PINPads com suporte a rede

Para a realização das transações com cartões com chip nos PINPads deve-se garantir que o PINPad saiba tratar estes cartões do Banrisul. Os PINPad com a versão da especificação da biblioteca compartilha 1.06a já suportam, porém aqueles com a versão 1.05g pode não suportar. Nestes casos deverá ser efetuada a transação com a tarja magnética do cartão. Para saber se o PINPad suporta o tratamento do cartão com chip Banrisul, deve-se entrar em contato do o banco Banrisul.

Associação dos produtos e as funções do SCOPE

Abaixo está uma tabela onde são relacionados os produtos do Banrisul e as funções da API do SCOPE que deve ser invocada para realizar a transação.

Produto Banrisul	Função da API do SCOPE
Convênio Refeisul combustível	`ScopeCompraCartaoDebito`
Cancelamento	`ScopeCancelamento`
Pagamento à vista, à prazo e parcelado	`ScopeCompraCartaoDebito`
Crédito 1 minuto	`ScopeCompraCDC`
Banco SIM parcelado	`ScopeCompraCartaoDebito`
Convênio Refeisul alimentação e refeição	`ScopeCompraCartaoDebito`
Convênio Private label	`ScopeCompraCartaoCredito`
Consulta de saldo de convênio Refeisul alimentação/refeição	`ScopeConsultaDebito`

Transação convênio combustível

Durante o fluxo de coleta desta transação, surgirão coletas específicas para a realização desta com o Banrisul, que são:

- Código de serviço: este é um código de 2 dígitos que informa o tipo de serviço realizado previsto pelo Banrisul. Os possíveis códigos e seus significados são:

- 01 - Gasolina
- 02 - Álcool
- 03 - Diesel
- 04 - GNV
- 05 - Combustível Aviação
- 06 - Óleo
- 07 - Lavagem
- 08 - Borracharia
- 09 - Óleo Marítimo
- 10 - Graxa
- 11 - Outros
- 12 - Gás GLP
- 13 - Filtro de Ar
- 14 - Filtro de Óleo
- 15 - Óleo Lubrificante
- 16 - Filtro de Combustível
- 17 - Gasolina Aditivada
- 18 - Serviço de Oficina

- Quantidade de serviço: é um valor numérico de até 12 dígitos que representa a quantidade do serviço prestado. Exemplo: se o código do serviço escolhido foi 03, então a quantidade solicitada é a quantidade de litros de diesel. Se o código foi 08, então é a quantidade de serviço de borracharia executado, que nesse caso poderia ser apenas 1.
- Matrícula: valor alfanumérico de até 11 dígitos.
- Hodômetro: valor numérico que é o hodômetro que está registrando no carro.
- Placa do veículo: campo numérico apenas de até 8 dígitos e deve ser entrada apenas a parte numérica da placa do carro.

Ticket Edenred

A implementação desta trata da especificação das transações com o cartão com chip Ticket Car, na modalidade Benefício e Gestão de Frotas. Transações com cartão são suportadas apenas utilizando o chip do cartão Ticket Car. As transações disponíveis no SCOPE que podem ser realizadas com esta rede são:

- Débito Voucher;
- Estorno Voucher;
- Consulta saldo;
- Atualização de parâmetros (Chip).
- Atualização de preços.

Para trabalhar com Gestão de Frotas é necessária configuração descrita na seção "Parametrização de Gestão de Frota" do "Manual de Instalação e Configuração" do Scope.

O SCOPE suporta a aprovação off-line, efetuada em POS, e uma característica deste cartão é que as transações aprovadas off-line são armazenadas no próprio chip do cartão para ser descarregado numa próxima transação realizada noutro estabelecimento. Portanto o SCOPE suporta o envio de transações realizadas em POS e gravadas no chip. Para a transação de envio destas off-line, é verdade que:

- Pode acontecer em qualquer uma das transações desta rede;
- Não é visível para o operador do caixa ou portador do cartão, ou seja, nenhum dos dois sabem que está enviando tais mensagens;
- Não é impresso comprovante para elas;
- Não é visto em relatórios do SCOPEADM;
- Apenas é possível verificar a existência do envio da offline pelos logs do SCOPE Server.

Não há transações com os cartões Ticket Alimentação e Ticket Restaurante roteadas para esta rede no SCOPE.

PINPads com suporte a rede

Para a realização das transações com cartões com chip nos PINPads deve-se garantir que o PINPad saiba tratar os cartões Ticket Car. Para isto deve-se utilizar PINPads Gertec modelo PPC900 com a versão da especificação da biblioteca compartilhada 1.07A (ou superior) contendo a chave Masterkey 3.

Associação dos produtos e as funções do SCOPE

Abaixo está uma tabela onde são relacionados os produtos da rede Ticket Edenred e as funções da API do SCOPE que deve ser invocada para realizar a transação.

Produto Ticket Edenred	Função da API do SCOPE
Compra online (Débito Voucher)	`ScopeCompraCartaoDebito` (vide item *Compra com cartão de débito*)
Estorno online	`ScopeCancelamento` (vide item *Estornando a transação*)
Consulta saldo	`ScopeConsultaSaldoDebito` (vide item *ScopeConsultaSaldoDebito*)
Atualização de parâmetros	`ScopeAtualizaParametrosChip` (vide item *ScopeAtualizaParametrosChip*)
Atualização de preços	`ScopeAtualizaPrecosMercadorias`

	(vide item *ScopeAtualizaPrecosMercadorias*)

Dados coletados

Abaixo está a relação do que pode ser solicitado em cada transação:

Transação	Coletas
Compra online (Débito Voucher)	• Cartão • Dados do ECF (veja mais abaixo) • Senha • Imprimir cupons • Dados Livres (Dados Adicionais)
Estorno online	• Número de controle da compra • Cartão • Imprimir cupons
Consulta saldo	• Cartão • Senha • Imprimir cupom do cliente
Atualização de parâmetros	• Cartão • Senha
Atualização de preços	• Nenhuma coleta

Consulta saldo

A função utilizada para esta transação é a `ScopeConsultaSaldoDebito` que como parâmetro espera receber um valor. No entanto, para os cartões Ticket Car não é utilizado este, portanto, a automação pode passar uma string vazia.
Esta transação não possui cupom da loja, mas apenas do cliente.

Dados do ECF

Na transação de compra online (débito Voucher) é necessário enviar dados do cupom e do ECF. O SCOPE retorna, no modo coleta, então o estado de coleta TC_COLETA_DADOS_ECF (FCBEh) e a automação deve invocar a função `ScopeForneceCampo` com os seguintes parâmetros:

- TypeField: tipo de dado:
  - SCOPE_DADOS_ECF_BENEFICIO → Fornecer os dados da impressora fiscal. A utilização desta constante deve ser feita na transação de débito, num estado de coleta TC_COLETA_DADOS_ECF. Informações adicionais na seção "Fornecendo informações extras para a transação"
- StructField: dados fornecidos pela aplicação, cujo formato para a constante utilizada na tabela abaixo.

Posição	Formato	Descrição
001-040	ans40	Número de série de fabricação do ECF (*alinhado à esquerda com brancos à direita*)
041-044	n4	Número do PDV atribuído ao ECF (*alinhado à direita com zeros à esquerda*)

045-064	n20	Número do documento fiscal (*alinhado à esquerda com brancos à direita*)
065-072	n8	Data de emissão do documento fiscal no formato AAAAMMDD.

Dados da Lista de Mercadorias Consumidas

Na transação de compra online (débito Voucher) é necessário enviar dados da Lista de Mercadorias Consumidas pela Aplicação. O SCOPE retorna, no modo coleta, o estado de coleta TC_COLETA_LISTA_MERCADORIAS (FCCBh) e a automação deve invocar a função `ScopeForneceCampo`(char TypeField, void *StructField), com os seguintes parâmetros:

- TypeField: tipo de dado:
  - SCOPE_DADOS_LISTA_MERCADORIAS → Fornecer os dados da Lista de Mercadorias Consumidas. A utilização desta constante deve ser feita na transação de débito, no estado de coleta TC_COLETA_LISTA_MERCADORIAS. Informações adicionais na seção "Fornecendo informações extras para a transação"
- StructField: dados fornecidos pela aplicação, cujo formato para a constante utilizada na tabela abaixo.

Posição	Formato	Descrição
001-004	an4	Header ("LM01") - Identificador do layout da lista
005-005	n1	Quantidade de Casas Decimais para Valor Unitário das mercadorias. Enviado no ID126 (Compreendido entre 0 e 7, inclusive)
006-006	n1	Quantidade de Casas Decimais para Quantidade das mercadorias. Enviado no ID145 (Compreendido entre 0 e 8, inclusive)
007-008	n2	Ramo Atividade "01" (Posto de Combust.) Código do Ramo de Atividade do Checkout. (*alinhado a direita e preenchidos com zeros à esquerda, caso necessário*)
009-013	an5	ID - "SC101" Contém identificador do formato do registro que será passado
014-016	n3	Qtd. Registros - Todos os registros a seguir deverão ser do mesmo formato (SC101): (*alinhado a direita e preenchidos com zeros à esquerda, caso necessário*)
017-106	ans90	Registro SC101 - Conforme descrição do formato.
107-196	ans90	Registro SC101 - Conforme descrição do formato.
...		

A consulta aos códigos das mercadorias está descrita posteriormente no ítem *Código das Mercadorias*

Descrição do formato do registro "SC101":

O formato deste registro foi baseado no formato do registro ID101 da especificação TicketCar, com o acréscimo da informação da descrição da automação para formatar o registro ID161 da TicketCar.

Posição	Formato	Descrição
001-040	an40	Descrição da Mercadoria da Automação Comercial.
041-045	n5	Código da Mercadoria consumida (Código TicketCar). (*alinhado a direita e preenchidos com zeros à esquerda, caso necessário*)
045-052	n8	Quantidade de Mercadoria consumida. (*alinhado a direita e preenchidos com zeros à esquerda, caso necessário*)
053-061	n9	Valor da Transação para a mercadoria consumida. (*alinhado a direita e preenchidos com zeros à esquerda, caso necessário*)
062-070	n9	Valor unitário ou valor unitário teto da mercadoria consumida (zerado, se transação for OFF-LINE). (*alinhado a direita e preenchidos com zeros à esquerda, caso necessário*)
071-079	n9	Valor da Transação para a mercadoria consumida (com desconto). (*alinhado a direita e preenchidos com zeros à esquerda, caso necessário*)
080-088	n9	Valor do desconto ($ ou %) Se (%) deverá ser enviado (x 100). Exemplo: 10,15% → 000001015 Se ($) deverá utilizar expoente de moeda da transação. Exemplo (expoente = 2): $ 15,66 → 000001566 (*alinhado a direita e preenchidos com zeros à esquerda, caso necessário*)
089-089	n1	Significado do campo "Valor do desconto (moeda ou percentual)". “0” = Valor ($) dado como desconto “1” = Valor (%) dado como desconto
Observação: Repetir este bloco tantas quantas forem as mercadorias que os preços foram atualizados.		

Dados da Lista de Atualização de Preços de Mercadorias

Na transação de Atualização de Preços de Mercadorias é necessário enviar dados da Lista de Atualização de Preços de Mercadorias. O SCOPE retorna, no modo coleta, o estado de coleta TC_COLETA_LISTA_PRECOS (FCCAh) e a automação deve invocar a função `ScopeForneceCampo` com os seguintes parâmetros:

- TypeField: tipo de dado:
    - SCOPE_DADOS_LISTA_PRECOS → Fornecer os dados da Lista de Atualização de Preços de Mercadorias. A utilização desta constante deve ser feita na transação de Atualização de Preços de Mercadorias, no estado de coleta TC_COLETA_LISTA_MERCADORIAS. Informações adicionais na seção "Fornecendo informações extras para a transação"

- StructField: dados fornecidos pela aplicação, cujo formato para a constante utilizada na tabela abaixo.

Posição	Formato	Descrição
001-004	an4	"LP01" Identificador do layout da lista
005-009	an5	"ID504" Contém identificador do formato do registro que será passado:
010-012	n3	Qtd. Registros - Todos os registros a seguir deverão ser do mesmo formato (ID504). (*alinhado a direita e preenchidos com zeros à esquerda, caso necessário*)
013-044	ans32	Registro ID504 - Conforme descrição do formato.
045-076	ans32	Registro ID504 - Conforme descrição do formato.
...		

Descrição do formato do registro "ID504":

Posição	Formato	Descrição
001-008	n8	Data da atualização de preço (Formato AAAAMMDD)
009-014	n6	Hora da atualização de preço (Formato HHMMSS)
015-019	n5	Código da mercadoria (*alinhado a direita e preenchidos com zeros àesquerda, caso necessário*)
020-031	n12	Valor do preço unitário (*alinhado a direita e preenchidos com zeros à esquerda, caso necessário*)
032-032	n1	Número de casas decimais para o valor unitário da mercadoria
Observação: Repetir este bloco tantas quantas forem as mercadorias que os preços foram atualizados.		

A consulta aos códigos das mercadorias está descrita posteriormente no ítem Código das Mercadorias.

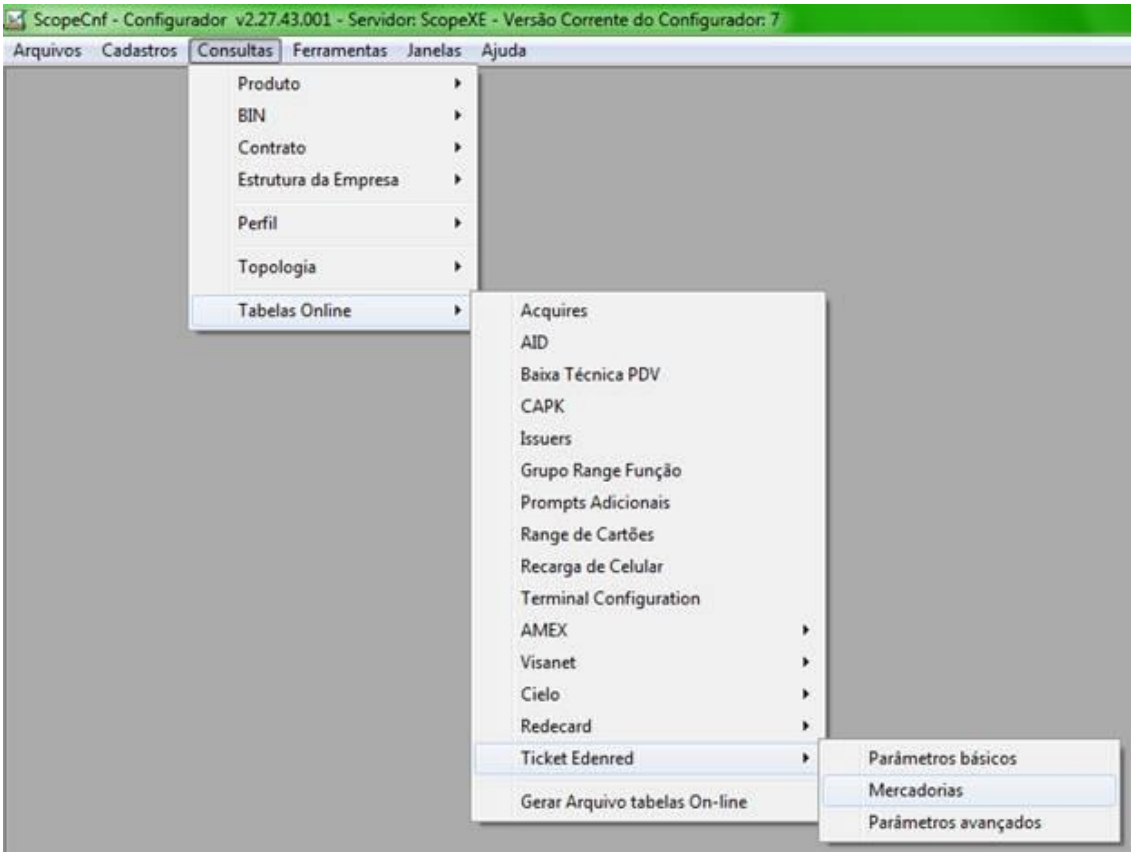
Comprovantes

Abaixo segue os comprovantes existentes para cada transação:

Transação	Coletas		
	Loja	Cliente	Reduzido
Compra online (Débito Voucher)	X	X	
Estorno online	X	X	
Consulta saldo		X	
Atualização de parâmetros (Dados Chip)			
Atualização de preços			

Código das Mercadorias

A Ticket Edenred envia Inicialização de Tabelas com as mercadorias e seus respectivos códigos. É possível consultar as mercadorias parametrizadas pela Ticket Edenred, através do ScopeCNF, acessando o menu exibido a seguir:



Cielo Premia

O CIELO Premia é um programa de premiação, através do qual, o comprador pode receber prêmios no momento da compra quando a TEF é transacionada pela Cielo. Também é conhecido como Plataforma Promocional Online da Cielo, ou, abreviadamente PP Online.

Como é a gestão da campanha?

Toda a gestão da campanha, inclusive sua temporalidade, é controlada pela autorizadora, ou seja, é a Cielo quem irá definir o período de validade da promoção, quais brindes serão oferecidos, regras de sorteio, cartões envolvidos, etc. As configurações são internas da Cielo e/ou Emissores e exigem uma atualização de tabelas (Inicialização). A atualização das tabelas é feita pelo SCOPE de forma online e automática, mediante sinalização da Cielo.

Quais tipos de prêmios o comprador pode receber?

Estão previstos os seguintes tipos de prêmios:

- Brindes, a serem resgatados posteriormente através de transação específica (Resgate de Prêmio); Chamado também de “resgate não monetário”;
- Desconto instantâneo, a ser aplicado no momento da compra, sobre o produto comprado. Chamado também de “resgate monetário”;

- Cupom eletrônico, que é apenas uma notificação de que o cliente está participando de uma campanha promocional. É chamado também de “e-cupom de participação”.

IMPORTANTE: Todos os tipos de prêmios podem ou não ocorrer simultaneamente em uma mesma transação.

Que tipos de transações podem receber os prêmios?

- Crédito;
- Débito;
- Voucher Alimentação;
- Voucher Refeição.

É possível restringir para tratar apenas prêmios não monetários, ou seja, não permitir descontos?

Caso o segmento de atuação ou os estados do Brasil em que sua automação é utilizada possuam restrições fiscais à concessão de descontos monetários, sugerimos consultar a empresa certificadora da CIELO e validar de antemão o tratamento que o tratamento que se pretende fazer na automação será adequado para a certificação.

O que é preciso configurar no SCOPE CNF?

É necessário habilitar o produto abaixo no Perfil de Serviços.

✔	Consulta Resgate de Premios	CIELO	CIELO

Bem como a bandeira CIELO na tela de Contrato:

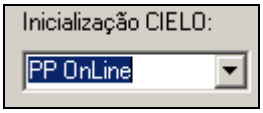
	Bandeira
⊠	CIELO

A Cielo precisa habilitar o Cielo Premia?

Sim, a Inicialização de Tabelas precisa contemplar tais produtos. O Lojista precisa entrar em contato com a CIELO e negociar a implantação do Cielo Premia.

É possível integrar e testar através do SCOPE Autorizador?

Sim. No ScopeAut há uma Inicialização de Tabelas que contempla o Cielo Premia, chamada “PP OnLine”, conforme mostrado abaixo:



IMPORTANTE:

1- Antes de iniciar os testes lembre-se de “forçar” uma Inicialização para o estabelecimento em questão para garantir que a carga “PP OnLine” seja carregada no banco de dados pelo ScopeSRV.

2- Os prêmios e cupons gerados pelo ScopeAUT possuem conteúdos fixos, e portanto, não representam efetivamente o que irá ocorrer em produção.

É possível fazer um teste usuando o HotKey?

Sim. O HotKeyHLAPI, HotKeyColeta e TstColeta estão preparados, porém deve ser configurada uma nova chave no SCOPE.INI do client, conforme exemplo abaixo (considerando empresa “0001” e filial “0001”):

```
[00010001]
AutomacaoParticipaPPOnline=S
```

A Automação Comercial deve se adequar?

Sim. A Automação Comercial precisa informar que está preparada para o Cielo Premia, adequar os fluxos de Crédito e Débito e permitir o Resgate de Prêmio.

O que a Automação Comercial precisa fazer para estar aderente?

Para habilitar o Cielo Premia, ou seja, configurar que está preparada:
Após o ScopeOpen, chamar a função ScopeForneceCampo passando como parâmetros o tipo de dado SCOPE_AUTOMACAO_PARTICIPA_PPONLINE e o conteúdo ‘S’ ou ‘1’ para indicar que SIM, ou seja, a automação está preparada para participar da plataforma Cielo Premia. Maiores detalhes sobre esta função estão no Manual do Desenvolvedor.
Note que o default (se não executar a função acima) é não habilitado. Neste caso a CIELO não irá retornar nenhum tipo de prêmio e tudo deve funcionar como antes.
Também é possível chamar a mesma função/dado passando ‘N’ ou ‘0’ para deixar o Cielo Premia desabilitado.
Enquanto não é executado o ScopeClose, o SCOPE mantém a informação se a AC está ou não participando do Cielo Premia válida para as próximas transações.
Também é possível que a AC defina se participa ou não do Cielo Premia a cada transação. Assim, pode-se chamar a ScopeForneceCampo passando SCOPE_AUTOMACAO_PARTICIPA_PPONLINE com SIM ou NÃO antes de cada ScopeCompraCartaoCredito e ScopeCompraCartaoDebito.

Para saber se a Cielo premiou com desconto monetário:
Tratar o novo código de retorno 263 (0x107) retornado pela ScopeStatus para indicar que a transação foi APROVADA COM DESCONTO. Caso o retorno da transação seja 000

(0x000), indica que a transação foi APROVADA (pode ou não ter recebido prêmio não monetário), mas com certeza, não houve desconto. Importante notar que ambos os retornos do ScopeStatus (000 e 263) indicam transação APROVADA.

Não é preciso esperar a finalização da transação para saber se houve ou não o desconto. Normalmente essa informação é necessária ao receber o cupom de TEF, para decidir como finalizar o cupom fiscal. A informação se houve ou não desconto e o valor do desconto podem ser obtidas nos estados de coleta de impressão de cupom.

IMPORTANTE: Quando há desconto na transação, o valor com desconto será exibido no pinpad para o cliente visualizar o valor final. Mensagem "VALOR COM DESCONTO".

Para saber o valor do desconto:

Para obter o valor do desconto, usar a função ScopeObtemCampoExt2, máscara 3, "Informação do desconto do resgate monetário" (0x00000002). Esta informação pode ser solicitada antes da impressão/solicitação do Cupom (ScopeGetCupomEx). Caso o valor retornado seja zero, então não houve desconto.

Como tratar o desconto no Cupom Fiscal:

Veja fluxo sugerido pela Cielo no documento "FLUXO BASICO CIELO PREMIA.pdf".

Como saber se houve um prêmio não monetário (brinde) recebido durante uma compra?

Se houve algum prêmio não monetário (um brinde), esta informação pode ser recuperada através da função ScopeObtemCampoExt2, máscara 3, "Informações sobre a quantidade e e-cupons disponíveis" (0x00000001). A informação segue a seguinte estrutura:

```
typedef struct {
char QtdeOcorrenciasECupom[2];
char DadosResgate[3000];
} stPPResgateNaoMonetario
```

Esta informação pode ser solicitada antes da impressão/solicitação do Cupom (ScopeGetCupomEx).

Os brindes serão acumulados pela Cielo através do número de cartão, portanto, o cupom da promoção é apenas informativo, não necessário para o resgate.

Note que, durante os fluxos de compra (crédito e débito), a informação se ocorreu um prêmio não monetário pode não ser relevante para automação, já que não há nenhum tratamento a fazer.

IMPORTANTE: Para cupom de tamanho zero, descartar o prêmio.

Como fazer um Resgate de Prêmio?

O resgate é feito no SCOPE através da função ScopeMenu. Esta função irá exibir os produtos disponibilizados pela Cielo para o chamado “Menu Outros” e poderá conter outras opções além do resgate de prêmio.

O operador deverá selecionar a opção correspondente, que irá acionar no SCOPE o serviço Consulta Resgate de Prêmio.

O fluxo do Resgate de Prêmio já prevê uma consulta automática à Cielo para buscar os prêmios disponíveis para um determinado cartão, lido no início do fluxo.

Para obter a lista de brindes disponíveis, AC pode tratar o estado de coleta TC_EXIBE_MENU_RESGATE_PREMIO retornado pela função ScopeStatus.

IMPORTANTE: Para que a opção “Resgate de Prêmio” esteja disponível no Menu, a inicialização de tabelas do estabelecimento em questão deverá estar atualizada de forma a contemplar o Cielo Premia.

Para os resgates não monetários, deve ser seguido basicamente o fluxo:

1. Recuperar do SCOPE os dados dos prêmios e exibir;
2. Coletar o prêmio escolhido e a quantidade;

Para recuperar e exibir os dados dos prêmios:

1. Tratar o estado TC_EXIBE_MENU_RESGATE_PREMIO;
2. Chamar a função ScopeObtemCampoExt2 pedindo pelos dados de resgate não monetário;
3. Exibir sequencialmente os prêmios de acordo com exemplo abaixo.

Exemplo:

```
char            aux[5000];
...
switch (RC)
{
case TC EXIBE MENU RESGATE PREMIO:

h = ScopeObtemHandle(0);
if (h > 0xFFFF)
{

ScopeObtemCampoExt2(h, 0x00, 0x00, Dados Resgate Nao Monetario, '\0', aux);
                                demo_ExibeResgateNaoMonetario(aux);

}
...
}
```

Onde “aux” contém um buffer que contenha espaço necessário para ser armazenado o número de prêmios e os dados relativos aos prêmios. O buffer será preenchido com a seguinte estrutura:

```
typedef struct {
char QtdeOcorrenciasECupom[2];
char DadosResgate[3000];
} stPPResgateNaoMonetario
```

Os dois primeiros bytes do buffer correspondem à quantidade de prêmios e os bytes restantes serão os dados dos prêmios. Para cada prêmio, ele será prefixado com 2 bytes que representam o tamanho do texto:

Posição	Formato	Descrição
01-02	N2	Número de E-cupons
03-04	N2	Tamanho do 1º E-cupom
05-x	A(n)	Título do 1º Prêmio, de tamanho n variável definido nos 2 bytes anteriores
...	...	Obs. Repetem-se tamanho e título para os N cupons

IMPORTANTE: A quantidade de prêmios pode ser igual a zero e corresponde que não há premiação.

Exemplo:

```
        void demo ExibeResgateNaoMonetario (char *aux)
        {

        stPPResgateNaoMonetario       sDadosResgateNaoMonetario;
        short i, nTamanhoEcupom = 0, j = 0, nQuantidade = 0;
        char  sTamanhoEcupom[2+1], sNomeEcupom[99+1], sQuantidade[2+1];

        memset (&sDadosResgateNaoMonetario, 0, sizeof(sDadosResgateNaoMonetario));
        memset (&sNomeEcupom, 0, sizeof(sNomeEcupom));
        memcpy(sQuantidade, aux, 2);
        sQuantidade[2] = 0;
        strcpy(sDadosResgateNaoMonetario.DadosResgate, &aux[2]);

        nQuantidade = atoi(sQuantidade);

        if (nQuantidade > 0)
        {
            printf ("\n\nPLATAFORMA PROMOCIONAL:\n\n");
        }
        else
        {
            printf ("\n\nNAO HA PREMIACAO, CONSULTE REGULAMENTO");
            return;
        }

        for(i = 0; i < nQuantidade && i < 20; i++)
        {
           memcpy(sTamanhoEcupom, &sDadosResgateNaoMonetario.DadosResgate[j], 2);
           sTamanhoEcupom[2] = 0;
           nTamanhoEcupom = atoi(sTamanhoEcupom);
           j += 2;
           memcpy(sNomeEcupom, &sDadosResgateNaoMonetario.DadosResgate[j],
nTamanhoEcupom);

            sNomeEcupom[nTamanhoEcupom] = 0;

            j += nTamanhoEcupom;

            printf ("%02d - %s", i + 1, sNomeEcupom);

            printf ("\n");
        }

        printf ("\n\n");

        return;
        }
```

Após a exibição dos prêmios, a automação pode fazer:

A coleta do índice/número do prêmio escolhido pelo cliente (TC_NRO_RESGATE_PREMIO 0xFCA4)

A coleta de quantidade a ser resgatada (TC_CONFIRMA_OPCAO_RESGATE_PREMIO 0xFCA2)

IMPORTANTE: : A quantidade de prêmios não é repassada para a Automação Comercial, somente é exibida no display.

Existe uma forma de tratar os itens do Menu Dinâmico pela aplicação de AC?

Sim. Os itens exibidos pela função ScopeMenu podem ser manipulados pela AC. Para isso existem as funções:

- ScopeMenuRecuperaItens;
- ScopeMenuSelecionaItem.

Através da função ScopeMenuRecuperaItens a AC pode tratar o estado TC_EXIBE_MENU para obter os itens de forma a exibi-los todos de uma só vez, ou então, exibir apenas os que interesse para a Automação Comercial.

A seleção também pode ser controlada através da ScopeMenuSelecionaItem. A Automação Comercial pode usar essa função após a seleção do operador de caixa, ou ainda, selecionar um item de forma automática, sem exibir a lista ao operador.

Existe um código exemplo?

Sim. Um código em C de exemplo (TstColeta) acompanha este documento.

Qual é a versão mínima do SCOPE que contempla Cielo Premia?

A versão mínima é 2.27.23.02 (server e client), embora o recomendável seja o uso da versão mais recente.

Cielo Auto

Esta transação é efetuada com cartões do tipo voucher que permitem o pagamento do abastecimento de combustível e outros serviços prestados aos veículos automotivos em estabelecimentos como postos de gasolina.

Na especificação CIELO – Release 2014, corresponde ao Fluxo 32 -“Transação Auto”. Transações são suportadas apenas utilizando o chip do cartão. A bandeira que irá tratar esta transação será:

- 278 - CIELO AUTO

As transações disponíveis no SCOPE que podem ser realizadas com este cartão são:

- Débito Voucher;
- Estorno Voucher;
- Consulta saldo;

- Atualização de parâmetros (Chip).

Lista de produtos e/ou serviços principais Cielo Auto

Esta lista poderá conter os seguintes ramos principais:

- COMBUSTÍVEL
- OUTROS PRODUTOS E SERVIÇOS
- ESTACIONAMENTO

Esta lista será exibida de acordo com as informações enviadas na Carga de Tabelas da CIELO e nas informações habilitadas no chip do cartão. O SCOPE disponibiliza os itens do Menu para a aplicação de automação comercial para a melhor exibição dessas informações. Esta lista é fornecida durante o fluxo de coleta através do estado abaixo:

TC_COLETA_RAMO_PRINC_CIELO_AUTO	0xFCD9	Coleta Ramo Principal Cielo Auto

Num fluxo de Compra com Cartao de Debito, recebendo o código de coleta 64729 (TC_COLETA_RAMO_PRINC_CIELO_AUTO), a aplicação de PDV pode realizar o mesmo tratamento do estado TC_EXIBE_MENU (que já existe hoje), ou seja, opcionalmente chamar a função ScopeMenuRecuperaItens do SCOPE para receber os itens do Menu a serem exibidos e em seguida chamar a ScopeMenuSelecionaItem para indicar ao SCOPE qual item foi selecionado.

Desta forma, o PDV poderá, por exemplo:

- Exibir todos os itens na mesma tela (hoje, pelo fluxo do SCOPE Client, é apresentado um item por vez);
- Excluir algum item da lista que eventualmente não deva ser exibido ao operador;
- Alterar a descrição de algum item para melhor entendimento do operador;
- Fazer a seleção automática de um determinado item sem apresentá-lo ao operador;

Caso o aplicativo de automação comercial opte por não tratar este estado de coleta, o SCOPE exibirá os itens na forma de menu rotativo.

Vale ressaltar que este menu somente será exibido se existir mais que um ramo principal a ser selecionado. Caso exista apenas um item, o SCOPE assume-o automaticamente sem exibir o menu.

Lista de produtos e/ou serviços secundarios Cielo Auto

Cada ramo principal poderá conter sua própria lista de produtos e/ou serviços secundários que atuam como **mercadorias** para automação comercial. Esta lista será exibida de acordo com as informações enviadas na Carga de Tabelas da CIELO e nas informações habilitadas no chip do cartão. Esta lista poderá conter as seguintes mercadorias:

COMBUSTÍVEL

Gasolina comum
Etanol
Diesel
Gasolina Aditivada
Gasolina Premium
Etanol Aditivado
Diesel S50
Diesel Aditivado
GNV (m3)
Biodiesel
OUTROS PRODUTOS E SERVIÇOS
Lava-Rápido
Aditivos e Lubrificantes
Filtros
Borracharia
Manutenção Geral
Equipamentos elétricos
ESTACIONAMENTO
Estacionamento

Esta lista é fornecida durante o fluxo de coleta através do estado abaixo:

TC_COLETA_COD_MERC_CIELO_AUTO	0xFCDA	Coleta Codigo de Mercadoria Cielo Auto

Caso exista apenas uma mercadoria habilitada na Carga de Tabelas ou no cartão, o SCOPE assume automaticamente essa mercadoria.

Num fluxo de Compra com Cartao de Debito, recebendo o código de coleta 64730 (TC_COLETA_COD_MERC_CIELO_AUTO), a aplicação de PDV deverá obter a lista de mercadorias disponíveis, através da função ScopeRecuperaBufTabela ().

Protótipo

```
LONG EXPORT ScopeRecuperaBufTabela (BYTE _TipoTabela,
                                    char *Qtd,
                                    char *ListaBuffer,
                                    WORD TamLista)
```

Descrição

- Esta rotina servirá para disponibilizar para a aplicação de automação comercial, uma lista de informações a serem exibidas na tela para facilitar a seleção da opção desejada.
- Ela servirá para facilitar futuras implementações que atendam a mesma finalidade.
- O parâmetro _TipoTabela indicará o formato dos dados que serão fornecidos.

Parâmetros

[out]	String [2+1]	Qtd	Retorna a quantidade de elementos da lista.
[out]	String	ListaBuffer	Retorna os registros da lista.
[in]	WORD	TamLista	Tamanho, em bytes, do campo “ListaBuffer”

Retorno

Ver tabela de código de retorno.

Parâmetro TipoTabela:

ID layout	Usar estrutura	Descrição
BUF_TAB_MERCADORIA_CIELOAUTO (ID 3)	stREG_MERCADORIA_CIELOAUTO	Contém os campos, conforme a transação AUTO da especificação CIELO .

Parâmetro ListaBuffer:

O SCOPE está preparado para tratar MAX_MERCADORIA_CIELOAUTO=72 mercadorias da CIELO AUTO.

A lista é formada por registros stREG_MERCADORIA_CIELOAUTO com os seguintes campos:

Posição	Formato	Descrição
01 a 02	char	Código do Ramo Principal.
03 a 04	char	Código da Mercadoria.
05 a 45	String	Descricao da Mercadoria.

Exemplo

```
int i = 0;
char qtd [2+1] = {0};
LONG lRet;
stREG_MERCADORIA_CIELOAUTO lista [MAX_MERCADORIA_CIELOAUTO] = {0};

lRet = ScopeRecuperaBufTabela (BUF_TAB_MERCADORIA_CIELOAUTO,
             qtd,  (char*) lista,
             sizeof (lista));

if (lRet == RCS_SUCESSO)
```

```
{
   // Sucesso no recebimento da lista de mercadorias
   for (i = 0; i < atoi(qtd); i++)
   {
       // exibe os membros das estruturas que estarão preenchidos
       // lista[i].CodRamoPrinc,
       // lista[i].CodMercadoria,
       // lista[i].Descricao
   }
}
else
    // Erro ...
```

Informações Adicionais Cielo Auto

Estas informações serão exibidas para coleta de acordo com a parametrização da Carga de Tabelas da CIELO e de acordo com as informações habilitadas no chip do cartão.

Informações Adicionais
Captura da placa
Hodômetro
Horímetro
Código do Motorista

A aplicação de PDV não necessita realizar qualquer tratamento para a coleta desses campos pois o SCOPE exibe de forma dinâmica, tanto a mensagem quanto os dados coletados, dependendo do cartão utilizado.

SAVS

Esta seção trata da especificação das transações com a rede SAVS (Sistema Autorizador de Vouchers e Serviços).

As transações implementadas no SCOPE que podem ser realizadas com esta rede são:

- Autorização de Voucher;
- Estorno de Autorização Voucher;

Associação dos produtos e as funções do SCOPE

Abaixo está uma tabela onde são relacionados os produtos da rede SAVS e as funções da API do SCOPE que deve ser invocada para realizar a transação.

Produto SAVS	Função da API do SCOPE
Autorização de Voucher	`ScopeServicosGenericos`
Estorno	`ScopeCancelamento`

Autorização de Vouchers e Serviços

Para a transação de Autorização de Vouchers e Serviços a automação deverá chamar a função ScopeServicosGenericos, passando como parâmetro uma string contendo o código de Serviço e uma string vazia.

Durante transação de Autorização de Voucher, como a rede SAVS trabalha com Carga de Tabelas, algumas coletas podem ocorrer, ou não, de acordo com a configuração enviada ao SCOPE na Inicialização de Tabelas.

O SCOPE seleciona internamente o fluxo de coleta, de acordo com a configuração enviada pela rede Autorizadora. Para facilitar o entendimento, seguem as coletas afetadas pela parametrização da Carga de Tabelas e a descrição dos Estados de Coleta que serão desviados pelo SCOPE:

- Parametrização de Fornecedor

De acordo com a parametrização do Fornecedor selecionado, as coletas solicitadas estão descritas na tabela a seguir:

- Coleta de Fornecedor:

Parâmetro da Carga		Coleta
Quantidade de Fornecedores Disponíveis no Segmento Selecionado	1	• Seleciona automaticamente o único Fornecedor disponível e pula para: • Coleta de Produto - TC_COLETA_PRODUTO_SAV (0xFCD6)
	>1	• Coleta de Fornecedor - Estado TC_COLETA_FORNECEDOR_SAV (0xFCD5)

- Coleta de códigos, valor, quantidade e Cliente Preferencial:

Parâmetro da Carga		Coleta
Tipo de coleta de dados do Voucher	"0"	• Código do Produto - Estado TC_COLETA_PRODUTO_SAV (0xFCD6) • Valor – Estado TC_COLETA_VALOR (0xFC34) ou TC_INFO_AGU_CONF_OP(0xFCFF) • Quantidade – Estado TC_COLETA_QUANTIDADE (0xFCD7)
	"1"	• Código EAN - Estado TC_COLETA_PRODUTO_SAV (0xFCD6) • Valor - EstadoTC_COLETA_VALOR (0xFC34) ou TC_INFO_AGU_CONF_OP(0xFCFF)
	"2"	• Código do Produto - Estado TC_COLETA_PRODUTO_SAV (0xFCD6) • Valor - EstadoTC_COLETA_VALOR (0xFC34) ou TC_INFO_AGU_CONF_OP(0xFCFF) • Número do Voucher - Estado TC_COLETA_NRO_VOUCHER (0xFCA5)
	"3"	• Código EAN - Estado TC_COLETA_PRODUTO_SAV (0xFCD6) • Valor - EstadoTC_COLETA_VALOR (0xFC34) ou TC_INFO_AGU_CONF_OP(0xFCFF) • Número do Voucher – Estado TC_COLETA_NRO_VOUCHER (0xFCA5)
Cliente preferencial	“0”	• Não solicita Coleta de Cliente Preferencial
	“1”	• Coleta: Cliente Preferencial? (Sim/Não) - Estado TC_COLETA_CLIENTE_PREFERENCIAL (0xFCD8)

Para cada Segmento selecionado no Estado de Coleta TC_COLETA_SEGMENTO_SAV (0xFCD4) podem estar disponíveis alguns Fornecedores. A lista de Fornecedores associada ao segmento selecionado, em passo anterior do fluxo de coleta, está descrita posteriormente neste documento na seção “Lista de Fornecedores”.

- Parametrização de Produtos

De acordo com a parametrização do Produto (Produto, Serviço ou Código EAN) selecionado podem ser habilitados:

- Coleta de dados dinâmicos (Numérico, alfabético, alfanumérico, menu de opções, Trilha de Cartão, Data, etc.) - Estado TC_COLETA_DADOS_ADICIONAIS (0xFC7C)
- Exibição de valor sugerido - Estado TC_COLETA_VALOR (0xFC34)

- Confirmação do Valor Assumido por Parametrização – Estado TC_INFO_AGU_CONF_OP (0xFCFF)

Para cada Fornecedor selecionado no Estado de Coleta TC_COLETA_FORNECEDOR_SAV (0xFCD5) podem estar disponíveis alguns Produtos. A lista de Produtos associada ao Fornecedor selecionado, em passo anterior do fluxo de coleta, está descrita posteriormente neste documento na seção “Lista de Produtos”.

Estorno de Autorização de Vouchers e Serviços

Para estorno da transação de Autorização de Vouchers e Serviços a automação deverá chamar a função ScopeCancelamento passando como parâmetro o valor total da transação original.

Lista de Fornecedores

A lista de fornecedores pode ser retornada para a automação comercial no novo layout, como descrito a seguir:

Num fluxo de Autorização de Serviços Voucher, recebendo o código de coleta 64725 (TC_COLETA_FORNECEDOR_SAV), a aplicação deverá obter a lista de fornecedores disponíveis, através da função ScopeRecuperaBufTabela ().

Protótipo

```
LONG EXPORT ScopeRecuperaBufTabela (BYTE _TipoTabela,
                                    char *Qtd,
                                    char *ListaBuffer,
                                    WORD TamLista)
```

Descrição

- Esta rotina servirá para disponibilizar para a aplicação de automação comercial, uma lista de informações a serem exibidas na tela para facilitar a seleção da opção desejada.
- Ela servirá para facilitar futuras implementações que atendam a mesma finalidade.
- O parâmetro _TipoTabela indicará o formato dos dados que serão fornecidos.

Parâmetros

[out]	String	Qtd	Retorna a quantidade de elementos da lista.
[out]	String	ListaBuffer	Retorna os registros da lista.
[in]	WORD	TamLista	Tamanho, em bytes, do campo “ListaBuffer”

Retorno

Ver tabela de código de retorno.

Parâmetro TipoTabela:

ID layout	Usar estrutura	Descrição
BUF_TAB_FORNECEDORES_SAVS (ID 1)	stREGISTRO_FORNECEDORES_SAVS	Fornecedores SAVS. Contém os campos, conforme a especificação do Autorizador Voucher.

Parâmetro ListaBuffer:

A lista é formada por registros stREGISTRO_FORNECEDORES_SAVS com os seguintes campos:

Posição	Formato	Descrição
01 a 04	String	Código do Fornecedor.
05 a 25	String	Nome do Fornecedor.
26 a 27	String	Código do Segmento de Mercado.
27 a 27	String	Quantidade máxima de produtos diferentes.

Exemplo

```
char sQtdFornec[3] = {0};
LONG lRet;
stREGISTRO_FORNECEDORES_SAVS lstFornecedores [20] = {0};

lRet = ScopeRecuperaBufTabela (BUF_TAB_FORNECEDORES_SAVS,
             sQtdFornec,  (char*) lstFornecedores,
             sizeof (lstFornecedores));

if (lRet == RCS_SUCESSO)
{
    /* Sucesso no recebimento da lista de fornecedores,
       os membros das estruturas estarão preenchidos */
}
else
    // Erro ...
```

Lista de Produtos

A lista de produtos pode ser retornada para a automação comercial no novo layout, como descrito a seguir:

Num fluxo de Autorização de Serviços Voucher, recebendo o código de coleta TC_COLETA_PRODUTO_SAV (0xFCD6), a aplicação deverá obter a lista de produtos disponíveis, através da função ScopeRecuperaBufTabela.

Protótipo

```
LONG EXPORT ScopeRecuperaBufTabela (BYTE _TipoTabela,
                                    char *Qtd,
                                    char *ListaBuffer,
```

WORD TamLista)

Descrição

- Esta rotina servirá para disponibilizar para a aplicação de automação comercial, uma lista de informações a serem exibidas na tela para facilitar a seleção da opção desejada.
- Ela servirá para facilitar futuras implementações que atendam a mesma finalidade.
- O parâmetro _TipoTabela indicará o formato dos dados que serão fornecidos.

Parâmetros

[out]	String	Qtd	Retorna a quantidade de elementos da lista.
[out]	String	ListaBuffer	Retorna os registros da lista.
[in]	WORD	TamLista	Tamanho, em bytes, do campo “ListaBuffer”

Retorno

Ver tabela de código de retorno.

ID layout	Usar estrutura	Descrição
BUF_TAB_PRODUTOS_SAVS (ID 2)	stREGISTRO_PRODUTOS_SAVS	Produtos SAVS. Contém os campos: **código**, **descrição e valores**, conforme a especificação do Autorizador Voucher.

Parâmetro ListaBuffer:

A lista é formada por registros stREGISTRO_PRODUTOS_SAVS com os seguintes campos:

Posição	Formato	Descrição
01 a 13	String	Código do Produto, Código do Serviço ou Código EAN.
14 a 34	String	Descrição.
35 a 47	String	Valor mínimo
47 a 59	String	Valor máximo
59 a 71	String	Valor sugerido

Exemplo

```
char sQtdProd[3] = {0};
LONG lRet;
stREGISTRO_PRODUTOS_SAVS lstProdutos [50] = {0};

lRet = ScopeRecuperaBufTabela (BUF_TAB_PRODUTOS_SAVS,
sQtdProd, (char*) lstProdutos, sizeof (lstProdutos));
```

```
if (lRet == RCS_SUCESSO)
{
    /* Sucesso no recebimento da lista de produtos,
       os membros das estruturas estarão preenchidos */
}
else
    // Erro ...
```

Cielo – Transações Sem Contato (Contactless)

São transações realizadas através de leitura sem contato no PINPad. Atualmente a Cielo 4.1 R2014, também conhecida como Cielo Release 2014, é a única que permite realizar esse tipo de transação.

Atualmente as bandeiras Visa, Visa Electron, Mastercard, Maestro e Amex são as únicas certificadas a operar através de leitura sem contato.

Requisitos

Para que esse tipo de transação seja aceito, o estabelecimento precisa estar trabalhando com Cielo 4.1 R2014. O PINPad precisa ser no mínimo 1.08a com módulo Contactless instalado e firmware atualizado. Em caso de dúvida, consulte o fabricante do PINPad para saber se seu PINPad está ou não preparado para aceitar Contactless.

Na inicialização de tabelas do estabelecimento em questão, a Cielo precisa habilitar e enviar na carga AIDs com Contctless habilitado.

O valor da transação precisa estar dentro da faixa permitida para Contactless, dado que também é configurado através da inicialização de tabelas. A carga no PINPad precisa estar atualizada com pelo menos um AID do grupo da transação em questão (Crédito ou Débito) com Contactless habilitado.

Funcionamento

A habilitação ou não da leitora Contactless é feita de forma automática, conforme tópicos descritos acima. Não é necessária nehuma configuração no SCOPE.

Quando ocorre erro na leitura sem contato, dependendo do tipo de erro ocorrido, a próxima tentativa pode desabilitar a leitura sem contato, de forma a evitar que o PINPAd leia novamente por aproximação quando na verdade, o portador do cartão deveria inserir o chip ou passar a tarja.

A leitura sem contato também é desabilitada automaticamente pelo PINPad quando o valor da trasação for acima do limite permitido para Contactless.

Existem dois tipos de cartão sem contato: “Chip EMV Sem Contato” e “Tarja Sem Contato”.

Transações de “Tarja Sem Contato” são equivalentes a uma transação “Magnética”.

Transações de “Chip EMV Sem Contato” são equivalentes a uma transação de “Chip EMV”.

A leitura “Sem Contato” é feita de forma única, ou seja, somente é necessário aproximar o cartão do PINPad uma única vez, mesmo quando se tratar do tipo “Chip EMV Sem Contato”.

Apêndice A – Tabelas

Códigos de retorno

Coleta de dados

Estes códigos serão retornados pela função ScopeStatus() quando a aplicação optar pela coleta dos dados através da função ScopeSetAplColeta(). Estes códigos informam para a aplicação de PDV a ação a ser tomada. Como exemplo, coletar um dado, imprimir um cupom e/ou cheque, ou até mesmo mostrar informações no “display” do operador e/ou cliente aguardando a confirmação.

Códigos Retorno		Significado
Hexa decimal	Decimal	
0xFC00	64512	Coletar cartão
0xFC01	64513	Coletar validade do cartão
0xFC02	64514	Imprime Cupom
0xFC03	64515	Coletar CGC ou CPF
0xFC04	64516	Coletar banco
0xFC05	64517	Coletar agência
0xFC06	64518	Coletar número do cheque
0xFC07	64519	Coletar data do cheque (bom para)
0xFC08	64520	Imprime cheque
0xFC09	64521	Coletar se a transação será a vista ou não
0xFC0A	64522	Coletar se a transação será parcelada pela administradora ou pelo estabelecimento
0xFC0B	64523	Coletar se a transação será pré-datada
0xFC0C	64524	Coletar se a parcela será à vista
0xFC0D	64525	Coletar quantidade de dias entre parcelas
0xFC0E	64526	Coletar quantidade de parcelas
0xFC0F	64527	Coletar o plano de financiamento
0xFC10	64528	Coletar o dia e o mês (DDMM)
0xFC11	64529	Coletar a senha
0xFC12	64530	Coletar o controle do SCOPE
0xFC13	64531	Coletar a forma de pagamento
0xFC14	64532	Coletar data do primeiro vencimento
0xFC15	64533	Coletar valor de entrada
0xFC16	64534	Coletar a forma de entrada
0xFC17	64535	Coletar conta corrente
0xFC18	64536	Coletar últimos dígitos do cartão
0xFC19	64537	Reimpressão de comprovante
0xFC1A	64538	Coletar se deseja consultar parcelas
0xFC1B	64539	Imprime consulta
0xFC1C	64540	Coletar decisão de continuar
0xFC1D	64541	Coletar decide último
0xFC1E	64542	Coletar número de cheque CDC
0xFC1F	64543	Coletar quantidade de dias
0xFC20	64544	Coletar o número da pré-autorização
0xFC21	64545	Coletar dia do mês fechado
0xFC22	64546	Imprime nota promissória

Códigos Retorno		Significado
Hexa decimal	Decimal	
0xFC23	64547	Coletar CEP
0xFC24	64548	Coletar número do endereço
0xFC25	64549	Coletar parte numérica do complemento
0xFC26	64550	Coletar plano de pagamento (Fininvest)
0xFC27	64551	Coletar ciclos a pular (Fininvest)
0xFC28	64552	Coletar número do item (Fininvest)
0xFC29	64553	Coletar código de segurança
0xFC2A	64554	Coleta se código de segurança ausente ou ilegível
0xFC2B	64555	Coleta se é com ou sem garantia de pré-datado
0xFC2C	64556	Coleta se aceita ou não risco
0xFC2D	64557	Coleta valor do saque
0xFC2E	64558	Coleta valor da recarga de celular pré-pago
0xFC2F	64559	Coleta código da localidade do telefone
0xFC30	64560	Coleta número do telefone
0xFC31	64561	Coleta dígito verificador do telefone
0xFC32	64562	Coleta data (formato DDMMAA)
0xFC33	64563	Coleta valor da taxa de serviço
0xFC34	64564	Coleta valor
0xFC35	64565	Coleta se quer realizar saque
0xFC36	64566	Coleta se quer realizar simulação de saque
0XFC37	64567	Coleta se quer saldo ou extrato
0xFC38	64568	Coleta se quer o extrato resumido ou a segunda via
0xFC39	64569	Coleta se é consulta investimento ou resgate
0xFC40	64576	Coleta se é resgate avulso
0xFC41	64577	Coleta data (formato DDMMAAAA)
0xFC42	64578	Coleta o código de autorização PBMS
0xFC43	64579	Coleta a lista de medicamentos
0xFC44	64580	Retorna a lista de medicamentos
0xFC45	64581	Exibir mensagem
0xFC46	64582	Imprime cupom parcial
0xFC47	64583	Coleta quantidade de parcelas e aceita 1 parcela
0xFC48	64584	Coleta código de barras
0xFC49	64585	Coleta código de consulta PBM
0xFC4A	64586	Coleta CRM médico
0xFC4B	64587	Coleta código UF CRM médico
0xFC4C	64588	Coleta se cliente deseja aderir ao seguro (IBICred)
0xFC4D	64589	Coleta se é pagamento com cartão
0xFC4E	64590	Coleta dados específicos da rede Tokoro
0xFC4F	64591	Coleta se deseja pagar após vencimento
0xFC50	64592	Coleta se a transação é com senha
0xFC51	64593	Imprime cupom promocional
0xFC52	64594	Coleta se utiliza saldo
0xFC53	64595	Coleta código do material
0xFC54	64596	Coleta código do plano
0xFC55	64597	Coleta se o pagamento é em cheque
0xFC56	64598	Coleta se confirma transação
0xFC57	64599	Coleta se o pagamento é no rotativo
0xFC58	64600	Coleta CMC7
0xFC59	64601	Coleta se o pagamento é em dinheiro ou TEF (cartão)
0xFC5A	64602	Coleta o código do grupo de serviço (para a TEF Externa)
0xFC5B	64603	Coleta o código da rede (para a TEF Externa)
0xFC5C	64604	Coleta o código do estabelecimento (para a TEF Externa)
0xFC5D	64605	Coleta o NSU do Host (para TEF Externa)
0xFC5E	64606	Coleta data (ddmmaaaa) (para TEF Externa)
0xFC5F	64607	Coleta se é consulta

Códigos Retorno		Significado
Hexa decimal	Decimal	
0xFC60	64608	Coleta se continua após verificar a conta
0xFC61	64609	Coleta o código da bandeira
0xFC62	64610	Coleta se é conta da fatura
0xFC63	64611	Coleta o valor total
0xFC64	64612	Coleta RG
0xFC65	64613	Coleta se deseja realizar novamente a transação (retentativa)
0xFC66	64614	Coleta somente CPF
0xFC67	64615	Coleta o endereço
0xFC68	64616	Coleta o andar
0xFC69	64617	Coleta o conjunto
0xFC6A	64618	Coleta o bloco
0xFC6B	64619	Coleta o bairro
0xFC6C	64620	Coleta a autorização ou o cartão
0xFC6D	64621	Coleta a data de emissão do cartão
0xFC6E	64622	Coleta o plano Infocards
0xFC6F	64623	Coleta número do cupom fiscal
0xFC70	64624	Coleta a operadora de recarga de celular pré-pago
0xFC71	64625	Coleta dados SAB
0xFC72	64626	Coleta o número do telefone com o dígito verificador
0xFC73	64627	Coleta os dados transação forçada SAB
0xFC74	64628	Coleta o tipo de serviço técnico: baixa técnica, teste comum ou estatística
0xFC75	64629	Coleta o número da OS
0xFC76	64630	Coleta identificação do técnico
0xFC77	64631	Coleta o código de ocorrência
0xFC78	64632	Coleta a EPS credenciada
0xFC79	*64633*	*Decide se coleta valor de entrada (não mais utilizado)*
0xFC7A	*64634*	*Decide se coleta valor da primeira parcela (não mais utilizado)*
0xFC7B	64635	Coleta o valor da primeira parcela
0xFC7C	64636	Coleta os dados adicionais
0xFC7D	64637	Coleta se cancela ou não a transação
0xFC7E	64638	Go On Chip
0xFC7F	64639	Retira o cartão
0xFC80	64640	Coleta o valor da taxa de embarque
0xFC81	64641	Exibe a mensagem de saldo
0xFC82	*64642*	*Exibir a mensagem e retorna o fluxo (não mais utilizado)*
0xFC83	*64643*	*Exibir a mensagem aguarda confirmação do operador (não mais utilizado)*
0xFC84	64644	Obtém os serviços
0xFC85	64645	Coleta o cartão digitado
0xFC86	64646	Coleta o código do produto
0xFC87	64647	Exibe o menu
0xFC88	64648	Coleta se é INSS ou cheque
0xFC89	64649	Coleta o contrato
0xFC8A	64650	Coleta a data quando o cliente aderiu ao cartão
0xFC8B	64651	Exibir o valor da consulta Vale Gás
0xFC8C	64652	Coleta data da transação original no formato DDMMAA
0xFC8D	64653	Coleta o NSU da transação original (número de 6 dígitos)
0xFC8E	*64654*	*Exibir os dados do cancelamento (não mais utilizado)*
0xFC8F	64655	Coletar qual é a via da reimpressão (0: todas as vias; 1: apenas da loja; 2: apenas do cliente)
0xFC90	64656	Coleta o DDD no PINPad
0xFC91	64657	Coleta o número de telefone no PINPad
0xFC92	64658	Coleta o número de telefone com o dígito verificador no PINPad
0xFC93	64659	Coleta a redigitação da recarga no PINPad
0xFC94	64660	Transação aprovada parcial
0xFC95	64661	Coleta o valor das parcelas

Códigos Retorno		Significado
Hexa decimal	Decimal	
0xFC96	64662	Coleta se a primeira parcela é para 30 ou 60 dias
0xFC97	64663	Uso interno do SCOPE e não retornado para a aplicação
0xFC98	64664	Uso interno do SCOPE e não retornado para a aplicação
0xFC99	64665	Uso interno do SCOPE e não retornado para a aplicação
0xFC9A	64666	Coleta o Número do Cartão (para a TEF Externa)
0xFC9B	64667	Coleta o código de serviço do posto (transação de convênio combustível Banrisul).
0xFC9C	64668	Coleta a matrícula (transação de convênio combustível Banrisul)
0xFC9D	64669	Coleta a quantidade do serviço (transação de convênio combustível Banrisul)
0xFC9E	64670	Coleta o hodômetro (transação de convênio combustível Banrisul)
0xFC9F	64671	Coleta a placa do veículo (transação de convênio combustível Banrisul)
0xFCA1	64673	Coleta o número do resgate de prêmio
0xFCA2	64674	Opção de resgate de prêmio selecionada
0xFCA3	64675	Solicitação a confirmação do resgate
0xFCA4	64676	Coleta o número do resgate prêmio
0xFCA5	64677	Coleta o número do Voucher
0xFCA6	64678	Decide entre DARF e GPS
0xFCA7	64679	Decide entre DARF Preto e Simples
0xFCA8	64680	Coleta o código de receita
0xFCA9	64681	Coleta o número de referência
0xFCAA	64682	Coleta o valor juros
0xFCAB	64683	Coleta o CPF do portador
0XFCAC	64684	Coleta o CNPJ
0XFCAD	64685	Coleta o percentual
0XFCAE	64686	Coleta competência
0xFCAF	64687	Coleta o número identificador
0xFCB0	64688	Coleta o valor INSS
0xFCB1	64689	Coleta a receita bruta
0xFCB2	64690	Coleta a opção do operador para continuar, alterar ou cancelar
0xFCB3	64691	Coleta o valor de acréscimo
0xFCB4	64692	Coleta o valor de dedução
0xFCB5	64693	Coleta redigitação do DDD
0xFCB6	64694	Coleta os 8 primeiros dígitos do cartão no Pin Pad
0xFCB7	64695	Coleta os 8 dígitos finais do cartão no Pin Pad
0xFCB8	64696	Coleta tipo da consulta (transação de Consulta Saldo Credito Infocards, sendo: 1=Saldo, 2=Dados Cadastrais)
0xFCB9	64697	Confirma o número do cartão digitado no Pin Pad
0xFCBA	64698	Coleta validação do saque
0xFCBB	64699	Coleta saque em andamento
0xFCBC	64700	Coleta DDD + Telefone no PIN Pad
0xFCBD	64701	Redigita DDD + Telefone no PIN Pad
0xFCBE	64702	Coleta dados ECF
0XFCC1	64705	Coleta opção de pagamento de carnê para GetnetLAC no serviço de débito a vista para bandeira ELECTRON (0=Não para pagamento de carnê, 1=Sim para pagamento de carnê).
0xFCC6	64710	Coletar dados cartão presente
0xFCCA	64714	Coleta da Lista de Atualização de Preços de Mercadorias
0xFCCB	64715	Coleta da Lista de Mercadorias Consumidas
0xFCCC	64716	Coleta de Horímetro
0xFCCD	64717	Coleta de Cartão Magnético
0xFCD4	64724	Coleta de Segmento de Mercado
0xFCD5	64725	Coleta de Código de Fornecedor de Produto ou Serviço
0xFCD6	64726	Coleta de Código de Produto, Código de Serviço ou Código EAN
0xFCD7	64727	Coleta de Quantidade

Códigos Retorno		Significado
Hexa decimal	Decimal	
0xFCD8	64728	Coleta de Confirmação de Cliente Preferencial
0xFCD9	64729	Coleta Ramo Principal Cielo Auto
0xFCDA	64730	Coleta Codigo de Mercadoria Cielo Auto
0xFCFC	64764	Coleta cartão em andamento
0xFCFD	64765	Coleta em andamento
0xFCFE	64766	Mostrar informações e retornar fluxo para o cliente SCOPE
0xFCFF	64767	Mostrar Informações e aguardar confirmação do operador

Autorizadoras

Abaixo se encontram relacionados os códigos devolvidos pelas redes autorizadoras na mensagem 0210 e entregue para a aplicação através da função ScopeStatus().

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x0000	0	Sucesso
0x0003	3	Estabelecimento comercial inválido
0x0006	6	Erro
0x0009	9	Transação em andamento
0x000C	12	Transação inválida
0x000D	13	Valor da transação inválido
0x000E	14	Cartão inválido
0x000F	15	Instituição não cadastrada
0x0013	19	Refaça transação
0x001E	30	Erro de formato
0x001F	31	Instituição não pertence à rede
0x0026	38	Excedido o número de tentativas do PIN
0x0029	41	Cartão extraviado
0x002B	43	Cartão roubado
0x0033	51	Saldo insuficiente
0x0034	52	Conta corrente não cadastrada
0x0036	54	Cartão vencido
0x0037	55	Senha incorreta
0x0038	56	Cartão sem registro
0x0039	57	Transação não permitida a esse cliente
0x003C	60	Entrar em contato com a instituição
0x003D	61	Excedido o limite de saque
0x0041	65	Excedida a frequência de saque
0x004C	76	Cartão bloqueado
0x004D	77	Pendente de confirmação
0x004E	78	Transação cancelada
0x004F	79	Transação não permitida neste ciclo
0x0050	80	Transação inexistente
0x0051	81	Transação estornada
0x0052	82	Chave de criptografia inválida
0x0053	83	Timeout
0x0054	84	Logon / Desfazimento
0x0055	85	Problema rede local
0x0056	86	Transação desfeita
0x0059	89	Mensagem enviada pelo Host

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x005A	90	Fechamento contábil
0x005B	91	Instituição temporariamente fora de operação
0x0100	256	Retorno genérico para códigos alfanuméricos. Para obter o exato código alfanumérico retornado é necessário chamar a função ScopeObtemCampoExt() passando o bit equivalente ao código de resposta (ver Obtendo os campos).
0x0101	257	Cliente com restrição na lista negra local
0x0102	258	Já consultou sob mesmo banco e agência
0x0103	259	Já consultou sob banco e/ou agência diferente(s)
0x0104	260	Código GAR não autorizado
0x0105	261	Compre Saque Redecard aprovado parcial
0x0106	262	Voucher Redecard aprovado parcial
0x0107	263	Informação de transação aprovada com prêmio

Códigos de erros do SCOPE

Em qualquer momento, na chamada de qualquer função o SCOPE Client devolverá um código de retorno. O código que não estiver na relação dos códigos de coleta e na das autorizadoras, encontram-se abaixo e geralmente denota um erro.

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFA01	64001	Parâmetro 1 inválido
0xFA02	64002	Parâmetro 2 inválido
0xFA03	64003	Parâmetro 3 inválido
0xFA04	64004	Parâmetro 4 inválido
0xFA05	64005	Parâmetro 5 inválido
0xFB01	64257	Não foi possível criar a "*thread*" na coleta de dados
0xFB02	64258	Erro na montagem do serviço pela API
0xFB03	64259	Erro ao verificar mensagem – mensagem inválida
0xFB04	64260	Erro ao montar mensagem
0xFB05	64261	Erro no arquivo de controle da TEF
0xFB06	*64262*	*Erro no contexto do arquivo de TEF (não mais utilizado)*
0xFB07	64263	Erro na totalização de TEF
0xFB08	64264	Erro no arquivo de controle utilizado finalização no ciclo multi-TEF
0xFB09	64265	Estourou o número máximo de TEF numa sessão multi-TEF
0xFB0A	64266	Não salvou a mensagem de confirmação para o SAB
0xFB0B	64267	Não salvou a mensagem de desfazimento para o SAB
0xFE00	65024	A transação em andamento – a aplicação deve aguardar
0xFE01	65025	SCOPE API não foi inicializada corretamente
0xFE02	65026	SCOPE API já foi inicializada corretamente
0xFE03	65027	Existe transação suspensa
0xFE04	65028	Não existe transação suspensa
0xFE05	65029	API ainda não fez nenhuma transação após a inicialização. A aplicação está tentando obter um *handle* sem ter feito nenhuma transação desde última conexão com o ScopeSRV
0xFE06	65030	Logon duplicado
0xFE07	65031	Protocolo não suportado – SCOPE Client com versão de protocolo superior ao SCOPE Server
0xFE08	65032	POS não cadastrado

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFE09	65033	Servidor não configurado no arquivo scope.ini
0xFE0A	65034	Não há mais PDV's disponíveis. Erro retornado quando a aplicação passa o caractere "#" como PDV para que o SCOPE Server lhe dê um número de PDV válido, mas não há mais PDV's disponíveis
0xFE0B	65035	Protocolo incompatível – SCOPE Client com versão de protocolo extremamente antigo (P00, P01 ou P02).
0xFE0C	65036	Não pode mais desfazer uma transação que já iniciou o ciclo de finalização – situação que pode ocorrer apos queda de energia ou na inicialização da aplicação.
0xFE0D	65037	Não há arquivo com dados da transação anterior salvo
0xFF00	65280	ScopeSrv *off-line* ou o IP configurado no arquivo scope.ini está apontando para a máquina errada
0xFF01	65281	Instituição *off-line*
0xFF02	65282	Transação cancelada pelo operador ou no caso de um estorno via REDECARD: estorno fora do prazo permitido, validade não confere.
0xFF03	65283	Serviço ou BIN não configurado
0xFF04	65284	Transação já foi cancelada
0xFF05	65285	Transação não encontrada
0xFF06	65286	Transação não permite cancelamento
0xFF07	65287	Dados informados não conferem com a transação original
0xFF08	65288	Erro no acesso ao banco de dados
0xFF09	65289	Time-out no acesso ao banco de dados
0xFF0A	65290	Banco de dados off-line
0xFF0B	65291	Transação abortada pelo aplicativo
0xFF0C	65292	Transação não implementada
0xFF0D	65293	*Handle* inválido (ver Obtendo handle)
0xFF0E	65294	Taxa de serviço é inválida
0xFF0F	65295	Taxa de serviço excede limite
0xFF10	65296	Dado inválido
0xFF11	65297	Não existe cupom válido
0xFF12	65298	Área reservada para o buffer é insuficiente para o SCOPE Client preencher com os dados solicitados
0xFF13	65299	Limite inválido – inferior ao permitido
0xFF14	65300	Transação desfeita
0xFF15	65301	Digitação não permitida
0xFF16	65302	Memória insuficiente
0xFF17	65303	"*Service Code*" inválido
0xFF18	65304	Data inválida
0xFF19	65305	Cartão vencido
0xFF1A	65306	Cartão inválido
0xFF1B	65307	Desfazimento não disponível
0xFF1C	65308	Erro na impressão do cupom
0xFF1D	65309	Sessão em andamento – a transação solicitada deve ser única numa sessão de TEF, ou seja, não pode ser chamada se há transações na sessão de TEF atual
0xFF1E	65310	Transação já efetuada
0xFF1F	65311	Inserir chip do cartão
0xFF20	65312	Controle obrigatório
0xFF21	65313	Pré-autorização obrigatória
0xFF22	65314	Serviço não configurado
0xFF23	65315	Serviço não definido
0xFF24	65316	Número de parcelas inválidas
0xFF25	65317	Valor inválido
0xFF26	65318	Serviço ou BIN não configurado para Visanet
0xFF27	65319	Estado de coleta não definido

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFF28	65320	Operação não permitida
0xFF29	65321	CNPJ/CPF inválido
0xFF2A	65322	Primeiro bloco do código de barras está inválido
0xFF2B	65323	Segundo bloco do código de barras está inválido
0xFF2C	65324	Terceiro bloco do código de barras está inválido
0xFF2D	65325	Quarto bloco do código de barras está inválido
0xFF2E	65326	Dados adicionais AID do chip do cartão inexistente
0xFF30	65328	Autorizador retornou dados inválidos
0xFF31	65329	Conta não permitida
0xFF32	65330	Conta vencida
0xFF33	65331	Não existe resumo
0xFF34	65332	Código de barra inválido
0xFF35	65333	Erro na consistência do DAC
0xFF36	65334	Erro no envio da confirmação ou desfazimento da transação anterior
0xFF37	65335	Serviço invertido – a aplicação está tentando fazer uma transação de débito com um cartão de crédito ou vice-versa
0xFF38	65336	Cartão não permitido
0xFF39	65337	Permitido somente a coleta do CPF – consulta de cheque
0xFF3A	65338	Erro interno na execução da coleta
0xFF3B	65339	Lista (de produtos, de medicamentos, etc.) não está disponível.
0xFF3C	65340	Erro de leitura do cartão
0xFF3D	65341	Controle inválido
0xFF3E	65342	Erro ao enviar mensagem para o servidor
0xFF3F	65343	Interface SAB não inicializada
0xFF40	65344	Erro: dados ainda não disponíveis (SAB)
0xFF41	65345	Erro: dados indisponíveis (SAB)
0xFF42	65346	Servidor SAB *off-line*
0xFF43	65347	Erro de conexão entre SCOPE e SAB
0xFF44	65348	Erro no NSU recebido
0xFF45	65349	Erro no logon do PDV
0xFF46	65350	Erro no processamento do chip
0xFF47	65351	Operadora inválida
0xFF48	65352	Dados, da recarga de celular, não encontrados.
0xFF49	65353	Transação cancelada pelo cliente
0xFF50	65360	Transação aprovada *off-line*
0xFF51	65361	Versão do banco de dados incompatível
0xFF52	65362	Cancelamento fora do prazo permitido
0xFF53	65363	Mensagem inválida
0xFF54	65364	PIN-Pad não foi aberto
0xFF55	65365	PIN-Pad já foi aberto – a aplicação abriu o PIN-Pad antes de conectar ao ScopeSRV, mas não deveria ter-lo feito, pois no ScopeCNF está configurado com uso exclusivo do SCOPE
0xFF56	65366	Estado inválido – a aplicação está tentando obter os serviços disponíveis num estado de coleta inadequado
0xFF57	65367	PIN-Pad compartilhado não está configurado, mas a rede exige que seja compartilhado
0xFF58	65368	PIN-Pad compartilhado não trabalha com a versão 2000 da VISA
0xFF59	65369	Função de uso exclusivo na interface coleta
0xFF5A	65370	Área insuficiente para os atributos do serviço
0xFF5B	65371	O SCOPE está configurado para uso de PIN-Pad compartilhado, mas a aplicação informou que está com o kit Visanet para PIN-Pad
0xFF5C	65372	O SCOPE não está configurado para uso de PIN-Pad compartilhado, mas a aplicação informou que está com PIN-Pad compartilhado
0xFF5D	65373	Erro ao inicializar periféricos – na interface HLAPI
0xFF5E	65374	Erro ao desmontar a estrutura ISO

Códigos Retorno		Significado
Hexadecimal	Decimal	
0xFF5F	65375	Bandeira não está configurada
0xFF60	65376	Função indisponível
0xFF61	65377	Valor mínimo da parcela é inválido
0xFF62	65378	Valor da consulta Vale Gás ainda não está disponível
0xFF63	65379	Número de Telefone inválido
0xFF64	65380	DDD inválido
0xFF65	65381	Erro Rede – Modelo 2
0xFF66	65382	Erro Rede – Modelo 3
0xFF67	65383	Confirmação Positiva Inconsistente
0xFF68	65384	Transação Offline – Permitido apenas reimpressão offline
0xFF69	65385	Contrato Suspenso
0xFF6A	65386	Transação permite somente digitado
0xFF6B	65387	Não Encontrado (para quando não veio menu dinâmico na carga de tabelas).
0xFF6F	65391	Erro ao acessar arquivo de contexto da PBM.
0xFFFF	65535	Erro genérico
-	-5000	Erro desmontando o pacote recebido
-	-5001	Não achou a Master Key referente
-	-5002	Ocorreu timeout do pacote
	-5003	Erro configurando o registrador
-	-5004	Parâmetro lib incorreto
-	-5005	Erro Leitura Cartao/Trilha

PIN-Pad compartilhado

Para alguns casos, o SCOPE poderá retornar erros reportados pelo PIN-Pad compartilhado, cujos códigos se encontram descritos na tabela abaixo.

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x0000	0	Operação OK
0x0001	1	Operação em processamento
0x0002	2	Notificar o operador com a mensagem recebida do PIN-Pad
0x0004	4	Pressionada a tecla de função #1
0x0005	5	Pressionada a tecla de função #2
0x0006	6	Pressionada a tecla de função #3
0x0007	7	Pressionada a tecla de função #4
0x0008	8	Pressionada a tecla “Limpar/Clear”
0x000A	10	Necessária operação previa
0x000B	11	Parâmetro inválido
0x000C	12	Time-out – esgotado o tempo de resposta
0x000D	13	Operação cancelada
0x000E	14	PIN-Pad já foi aberto
0x000F	15	PIN-Pad não foi aberto
0x0010	16	Erro interno de execução
0x0011	17	Função não suportada
0x0012	18	Função não disponível
0x0013	19	Ausência de dado mandatório para o processamento.
0x0014	20	Tabelas expiradas
0x0015	21	Erro ao tentar gravar tabelas (falta de espaço, por exemplo)

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x0016	22	Aplicação da rede adquirente não existe no PIN-Pad.
0x001E	30	Erro de comunicação: porta serial do PIN-Pad provavelmente ocupada.
0x001F	31	Erro de comunicação: PIN-Pad provavelmente desconectado ou problemas com a interface serial. Pode ser que a porta que está passando como parâmetro esteja errada.
0x0020	32	Status informado pelo PIN-Pad não é conhecido.
0x0021	33	Mensagem recebida do PIN-Pad possui formato inválido.
0x0022	34	Tempo esgotado ao esperar pela resposta do PIN-Pad (no caso de comandos não-blocantes).
0x0028	40	Erro interno do PIN-Pad.
0x0029	41	Erro de leitura do cartão magnético.
0x002A	42	Erro na captura do PIN (senha) - *Master Key* pode não estar presente.
0x002B	43	Não há cartão inteligente presente no acoplador. Cartão removido.
0x002C	44	PIN-Pad não pode processar a captura de PIN temporariamente devido a questões de segurança (como quando é atingido o limite de capturas dentro de um intervalo de tempo).
0x0032	50	Erro genérico no módulo SAM.
0x0033	51	SAM ausente, "mudo", ou com erro de comunicação.
0x0034	52	SAM inválido, desconhecido ou com problemas.
0x003C	60	Cartão não responde ("mudo") ou chip não presente.
0x003D	61	Erro de comunicação do PIN-Pad com o cartão inteligente.
0x003E	62	Cartão do tipo inválido ou desconhecido, não pode ser tratado (não é EMV nem TIBC v1).
0x003F	63	Cartão bloqueado por número excessivo de senhas incorretas (somente para Easy-Entry TIBC v1).
0x0040	64	Cartão TIBC v1 não autenticado pelo módulo SAM (somente para Easy-Entry TIBC v1).
0x0041	65	Cartão TIBC v1 expirado (somente para Easy-Entry TIBC v1).
0x0042	66	Cartão com erro de estrutura - arquivos estão faltando.
0x0043	67	Cartão foi invalidado. Se o cartão for TIBC v1, quando seleção de arquivo ou ATR retornar status ‘6284’. Se o cartão for EMV, quando seleção de aplicação retornar status ‘6A81’.
0x0044	68	Cartão com problemas. Esse status é válido para muitas ocorrências no processamento de cartões TIBC v1 e EMV onde o cartão não se comporta conforme o esperado e a transação deve ser finalizada.
0x0045	69	O cartão, seja TIBC v1 ou EMV, comporta-se corretamente, porém possui dados inválidos ou inconsistentes.
0x0046	70	Cartão sem nenhuma aplicação disponível para as condições pedidas (ou cartão é reconhecido como TIBC v1 ou EMV, mas não possui nenhuma aplicação compatível com a requerida).
0x0047	71	Somente para cartão EMV. A aplicação selecionada não pode ser utilizada neste terminal, pois o “*Get Processing Options*” retornou status ‘6985’.
0x0048	72	Somente para aplicação de moedeiro. O saldo do moedeiro é insuficiente para a operação.
0x0049	73	Somente para aplicação de moedeiro. O limite máximo para a operação foi excedido.
0x004A	74	Cartão ainda não efetivo (data de ativação posterior à atual)
0x004B	75	Moeda inválida (cartão moedeiro)
0x004C	76	Erro de alto nível no cartão EMV que é passível de “fallback” para tarja magnética.
0x0050	80	Mais de um cartão sem contato foi apresentado ao leitor (este código de retorno é opcional e depende da capacidade do equipamento em detectar esta situação).

Códigos Retorno		Significado
Hexadecimal	Decimal	
0x0051	81	Erro de comunicação entre o terminal (antena) e o cartão com chip sem contato.
0x0052	82	Cartão foi invalidado (seleção de aplicação retornou status ‘6A81’).
0x0053	83	Cartão com problemas. Esse status é válido para muitas ocorrências no processamento de cartões sem contato em que o cartão não se comporta conforme o esperado e a transação deve ser finalizada.
0x0054	84	Cartão sem nenhuma aplicação disponível para as condições pedidas (nenhum AID encontrado).
0x0055	85	A aplicação selecionada não pode ser utilizada (o Get Processing Options retornou status ‘6985’ ou houve erro no comando Select final), e não há outra aplicação compatível na lista de candidatas.
0x00C8	200	Transação negada na função PP_GoOnChip()
0x00C9	201	Transação negada na função PP_GoOnChip()
0x00CA	202	Memória não alocada para a estrutura do PIN-Pad compartilhado
0x00CB	203	Erro alocando memória
0x00CC	204	Memória insuficiente para receber os dados
0x00CD	205	PIN-Pad já aberto via SCOPE
0x00CE	206	Não foi possível definir a *Master Key* a ser utilizada
0x00CF	207	Não foi possível definir o Estado de coleta no PIN-Pad
0x00D0	208	Erro no parâmetro da função GetPIN
0x00D1	209	PIN-Pad não configurado
0x00D2	210	Display não permitido neste momento ou situação
0x00D3	211	PIN-Pad não foi aberto pela aplicação
0x00D4	212	Time-out do cliente / usuário
0x00D5	213	Dado no chip não encontrado
0x00D6	214	Comanda Vazia
0X00D7	215	Comanda Inválida

Códigos de Retorno de Pré-TEF

Os códigos abaixo podem ser retornados pelo ScopeSRV ao ScopeAPI quando ocorre erro em uma Pré-TEF. Embora sejam erros internos ao SCOPE podem ser úteis para identificar e ajudar na análise de logs para identificação de problemas:

Código de Retorno (Decimal)	Descrição
00	Sucesso
01	Rede Offline
02	Erro na consulta ao BD
03	Timeout na consulta ao BD
04	BD não conectado
05	Transação não localizada no BD
06	BIN ou Serviço não configurado
07	PDV já logado
08	PDV não cadastrado
09	Taxa de serviço inválida

10	Taxa de serviço excede limite
11	Digitação não permitida
12	Protocolo não suportado
13	Não há PDVs disponíveis
14	Protocolo incompatível
15	Inserir cartão chip
16	BIN ou serviço não configurado pela rede adquirente
17	Conta não permitida
18	Serviço “invertido” (Crédito por Débito ou vice-versa)
19	Identificação de PDV inconsistente (recebeu um buffer inválido)
20	Erro na leitura do cartão
21	Dados não encontrados
22	Rede inválida ou não suportada pelo PINPad desse PDV
23	Dados de recarga não encontrados
24	Versão do BD incompatível
25	Chip não habilitado pela rede
26	Serviço não permitido para chip
27	Chip sem dados adicionais (pode indicar falha na carga de tabelas)
28	Cartão com Service Code inválido
29	Rede exige uso de PINPad compartilhado
30	PINPad compartilhado não opera com rede VISANET 2000.01
31	Estorno fora do prazo permitido
32	Estorno parcial não permitido (valor deve ser o total)
33	Estorno excede valor máximo permitido
34	Estorno com valor acima do total original não permitido
35	Estorno com valor acima do total original excede limite permitido
36	Estorno com valor zerado não permitido
37	Estorno com valor original zerado não permitido
38	Estorno já realizado
39	Cupom não encontrado
40	Prompts Adicionais não encontrados (pode indicar falha na carga de tabelas)
41	Contrato suspenso
42	Cartão inválido
43	Modo inválido
44	Leitor inativo
45	Erro nas chaves de criptografia
46	Erro na parametrização de trilhas da rede REDE (pode indicar falha na inicialização de tabelas)

47	Erro na leitura da tarja do cartão (Trilhas inconsistentes ou fora do padrão. Pode indicar falha de leitura, cartão corrompido ou fraudado)
48	Falha na configuração da confirmação positiva da rede adquirente GetNetLac
49	Transação por celular não habilitada no contrato
50	Transação por celular exige que a rede habilite transação digitada
51	A rede está em processo de inicialização. Tente novamente.
52	PINPad não suportado em modo “Multi-Bandeira Estendido”
53	Não representa erro. Uso interno em modo “Multi-Bandeira Estendido”.
54	Erro de leitura. PAN excede tamanho máximo.
55	Erro de leitura. Trilha1 excede tamanho máximo.
56	Erro de leitura. Trilha2 excede tamanho máximo.
57	Contactless não habilitado na carga de tabelas
58	Chip Contactless sem dados adicionais (pode indicar falha na carga de tabelas)
59	Contactless sem produto habilitado na carga de tabelas
60	Excede valor limite definido para Contactless

Formatos dos dados

Utilizado na coleta do dado, a aplicação valida a entrada de dado que o usuário forneceu de acordo com o código do formato do dado recebido.

Códigos Formato		Significado
Hexadecimal	Decimal	
0x0000	00	String representando uma data no formato “DDMMAA”
0x0001	01	String representando uma data no formato “DDMM”
0x0002	02	String representando uma data no formato “MMAA”
0x0003	03	String representando uma hora no formato “HHMMSS”
0x0004	04	String representando um número
0x0005	05	String representando uma senha que é numérica
0x0006	06	String representando um número com 4 dígitos
0x0007	07	String representando um dado alfanumérico
0x0008	08	String representando uma data no formato “DDMMAAAA”

LEMBRETE: utilizado apenas com a interface coleta**.**

Códigos das Teclas

Durante o processamento da transação, a aplicação deverá disponibilizar um meio que permita que usuário prossiga, retorne ou cancele o processamento. No entanto, de acordo com o

momento do processamento, ou seja, conforme o estado da coleta de dados em que o SCOPE Client se encontra, nem sempre estas três opções estarão disponíveis. Para que a aplicação saiba qual(is) ação(ões) ela deve disponibilizar ao usuário, ela deve utilizar o membro HabTeclas da estrutura stPARAM_COLETA, após a chamada à função ScopeGetParam(). O valor deste campo é a combinação binária (bitwise) de:

Códigos Formato		Significado
Hexadecimal	Decimal	
0x0001	01	Tecla cancela habilitada
0x0002	02	Tecla próximo habilitada
0x0004	04	Tecla retorna habilitada

Ele deverá ser verificado a cada iteração, logo após a chamada à função (ver Obtendo os parâmetros da transação)

LEMBRETE: utilizado apenas com a interface coleta**.**

Códigos de Fluxo

Num determinado momento, coletado o dado ou não, a aplicação deverá informar ao SCOPE qual é ação que ele deverá tomar (ver Passando o dado da coleta ao SCOPE Client). Esta ação está associada às opções que o usuário pode tomar: avançar para o próximo estado, retornar para o estado anterior ou cancelar a transação. Cada ação tem um código amarrado a ela que se encontra na tabela que segue.

Códigos Formato		Significado
Hexadecimal	Decimal	
0x0000	00	Próximo estado
0x0001	01	Estado anterior
0x0002	02	Cancelar
0x0003	03	Erro na coleta de dados

LEMBRETE: utilizado apenas com a interface coleta**.**

Estados para interrupção

Na tabela abaixo estão relacionados os códigos referentes aos estados de coleta, nos quais o SCOPE poderá ser suspenso (ver Suspendendo a transação).

Índice	Exit Point
0	Coleta do cartão
1	Coleta da data de validade do cartão
2	Montagem e envio da mensagem
3	Exibição de mensagens de não autorização

4	Impressão do cupom
5	Recuperação dos serviços válidos
6	Coleta de CPF ou CGC
7	Resultado com transação cancelada
8	Resultado com transação autorizada
9	Coleta do número do banco
10	Coleta do número da agência
11	Coleta do número do cheque
12	Coleta da data para depósito
13	Impressão de cheque
14	Escolha de pagamento à vista
15	Escolha do plano de parcelamento da administradora de cartão ou da loja
16	Escolha de pré-datado
17	Escolha de parcela à vista
18	Coleta do intervalo de datas entre parcelas
19	Coleta de quantidade de parcelas
20	Decidindo o plano de financiamento
21	Coleta de número de dias ou da data de débito
22	Decide se coleta a senha
23	Coleta de senha
24	Carrega da bandeira
25	Desfazimento de transação
26	Recuperação de dados de transação
27	Coleta o número de controle
28	Consistência dos dados originais da transação (cancelamento)
29	Consistência do cartão
30	Montagem e envio da mensagem de pré-autorização
31	Escolha da forma de pagamento (1 - cheque, 2 – carnê, 4 - C/C)
33	Coleta do valor da entrada (Valor Entrada ?)
36	Coleta do cheque da parcela ("Num Cheque Entrada ?" ou "Num Cheque Parcela ?”)
37	Coleta do número da conta-corrente do cheque (Conta Corrente ?)
38	Encerramento da transação abortada
39	Decide se deve coletar os 4 últimos dígitos
40	Coleta os últimos 4 dígitos do cartão
41	Verificação do cartão e do serviço localmente
42	Decide se é a última transação
43	Recuperação do comprovante
44	Decide se consulta parcelas
45	Consulta de parcelas
46	Impressão de consulta
47	Decide de continua a transação
48	Exibe mensagem de erro
49	Exibe autorizado
50	Decide se coleta á vista
51	Decide se coleta o tipo de financiamento ou a quantidade de parcelas
52	Decide se é pré-datado ou parcelado
53	Decide se coleta se a parcela é á vista
54	Coleta cartão para resumo de vendas
55	Decide se continua ou termina
56	Exibe continua
57	Exibe fim
58	Coleta o número do cheque de entrada
59	Decide se imprime cheque
60	Coleta a quantidade de dias
61	Coleta a quantidade de dias para a primeira parcela
62	Coleta a data para a primeira parcela
63	Carrega os atributos para pré-datado
64	Carrega os atributos de parcela com a primeira à vista

65	Carrega os atributos dos dias entre parcelas
66	Carrega os atributos referentes às quantidades de parcela
67	Decide se coleta pré-autorização
68	Coleta pré-autorização
69	Coleta se confirma o dia do mês fechado
70	Carrega os atributos de parcela sem parcela à vista
71	Decide se coleta o cartão
72	Verifica o cheque
73	Imprime a nota promissória
74	Coleta a data da pré-autorização
75	Coleta o CEP
76	Coleta o número da residência
77	Coleta o complemento do endereço
78	Decide se consulta saque
79	Consulta saque
80	Decide se coleta o número do item
81	Coleta o plano de pagamento
82	Coleta ciclos a pular
83	Coleta o número do item
84	Decide se coleta o código de segurança
85	Coleta o código de segurança
86	Coleta a autorização do supervisor
87	Coleta se o código de segurança do cartão está ausente ou ilegível
88	Carrega os atributos de garantia pré-datado
89	Coleta se é com ou sem garantia
90	Decide se aceita risco
92	Decide se coleta o valor do saque
93	Coleta o valor do saque
94	Consulta valores
95	Coleta o valor de recarga de celular
96	Verifica o valor de recarga
97	Coleta o código de localidade do telefone
98	Coleta o número de telefone
99	Carrega a sugestão de recarga de celular
100	Verifica a transação
101	Coleta o dígito verificador do telefone
102	Decide se coleta a taxa de serviço
103	Coleta a taxa de serviço
104	Coleta data no formato DDMMAA
105	Decide se coleta data no formato DDMMAA
106	Decisão prévia do serviço
107	Decisão prévia do serviço de consulta
108	Decisão prévia do serviço de extrato
109	Decide se haverá saque
110	Decide se haverá simulação de saque
111	Decide se é saldo ou extrato
112	Decide o extrato é resumido ou segunda via
113	Decide se coleta valor
114	Coleta valor
116	Decide se coleta a quantidade de parcelas
117	Valida dados
118	Exibe mensagem de erro
119	Coleta se quer consultar ou resgatar
120	Decisão prévia do serviço de resgate
121	Coleta se é resgate avulso
122	Desfazimento de risco pré-datado
123	Decide se coleta data no formato DDMMAAAA
124	Coleta data no formato DDMMAAAA

125	Verifica mensagem de advertência
126	Exibe mensagem
127	Coleta código de autorização de medicamento
128	Coleta registro de medicamento
129	Obtém lista de medicamentos
130	Imprime cupom parcial
131	Coleta a quantidade de parcelas e aceita 1 parcela
132	Coleta o código de barras
133	Decide se coleta vencimento
134	Coleta vencimento
135	Coleta o código de consulta PBM
136	Coleta medicamento se receita
137	Coleta CRM do médico
138	Coleta a UF do CRM do médico
139	Decide se coleta seguro
140	Coleta seguro
141	Decide se coleta cartão
142	Coleta se é pagamento com cartão
143	Coleta dados da Tokoro
144	Decisão prévia de financiamento pela administradora
145	Decide se quer pagar após o vencimento
146	Decide se coleta a senha
147	Imprime o cupom promocional
148	Coleta se utiliza o saldo
149	Coleta o código de material
150	Coleta o número do plano
151	Coleta se o pagamento é em cheque
152	Coleta se confirma a transação
153	Verifica a não confirmação da transação
154	Decide se o pagamento é no rotativo
155	Confirma o valor
156	Exibe garantia não autorizada
157	Coleta CMC7
158	Coleta se o pagamento é em dinheiro ou cartão
159	TEF Externa: coleta o código do grupo de serviço
160	TEF Externa: coleta o código da rede
161	TEF Externa: coleta o código do estabelecimento
162	TEF Externa: coleta o NSU do host
163	TEF Externa: coleta a data no formato DDMMAAAA
164	TEF Externa: verifica a transação
165	Decide se quer realizar uma consulta antes
166	Consultando
167	Coleta se continua após a aprovação da conta
168	Coleta o código de bandeira
169	Coleta se é fatura
170	Verifica o código de barras
171	Coleta a confirmação do valor
172	Decide se imprime cupom total ou parcial
173	Coleta valor total
174	Coleta se quer resumo do pagamento
175	Coleta RG
176	Decide retentativa
177	Coleta CPF
178	Coleta endereço
179	Coleta andar
180	Coleta número do conjunto
181	Coleta o bloco
182	Coleta o bairro

183	Coleta código de autorização ou cartão
184	Coleta data da emissão do cartão
185	Coleta o plano - Infocards
186	Coleta o número do cupom fiscal
187	Coleta a operadora
188	Coleta dados SAB
189	Coleta o número do telefone com o dígito verificador
190	Coleta dados da transação forçada SAB
191	Decide o tipo de serviço técnico
194	Coleta o número da OS
195	Coleta o número de identificação do técnico
196	Coleta código da ocorrência
197	Coleta código da EPS credenciada
198	Go On Chip
200	Recupera as tabelas de PIN-Pad
201	Coleta confirmação positiva
202	Decide se coleta o valor da entrada
203	Decide se coleta o valor da 1ª parcela
204	Coleta o valor da 1ª parcela
205	Salva serviço escolhido
206	Decide se consulta parcelas ou envia TEF
207	Decide se coleta Chip ou senha
208	Decide se coleta dados adicionais
209	Coleta dados adicionais
210	Decide se carrega bandeira ou consulta operadora
211	Consulta operadora
212	Decide se confirma valor no PIN-Pad
213	Transação cancelada pelo cliente
214	Decide se cancela transação
215	Coleta se cancela a transação
216	Coleta em andamento
217	Retirar o cartão
218	Transação aprovada
219	Decide se coleta a taxa de embarque
220	Coleta a taxa de embarque
221	Coleta de cartão em andamento
222	Exibe mensagem de saldo
225	Coleta cartão digitado
226	Obtém serviços
227	Exibe saldo no PIN-Pad
228	Encerra transação por erro
229	Coleta o código do produto
230	Exibe mensagem recebida do PIN-Pad e retorna
231	Exibe mensagem de erro recebida do PIN-Pad e retorna
232	Exibe menu
233	Encerra menu
234	Encerra transação
235	Decide se é INSS
236	Coleta contrato
237	Verifica CPF
238	Coleta data de “*membership*” do cliente
239	Recupera comprovante do servidor
240	Fornece valor
241	Recupera menu de cancelamento
242	Coleta dados da transação original
243	Retorna dados da transação original
244	Coleta número da autorização original
245	Coleta data da transação original

246	Coleta NSU da transação original
247	Exibe dados de cancelamento
248	Decide as vias de reimpressão
249	Exibe mensagem rotativa
250	Coleta do DDD no PINPad
251	Coleta do número do telefone no PINPad
252	Coleta do número do telefone e dígito verificador no PINPad
253	Coleta de redigitação no PINPad
254	Transação aprovada parcial
255	Coleta valor das parcelas
256	Coleta 30 ou 60 dias para a cobrança da primeira parcela
257	Verifica se é CDC ou é Parcele Mais
258	Coleta se é à vista com juros
259	Transação cancelada por *fallback* inválido
260	Coleta a utilização do saldo restante do cartão voucher
261	Critica o valor na transação de Troco Surpresa
262	Consulta para transação de Vale-Gás
263	Verifica se deve coletar o código de ativação do celular
264	Coleta o código de ativação do celular
265	TEF Externa: Coleta Número do Cartão
278	Coleta código de serviço do posto
279	Coleta matrícula
280	Coleta a quantidade do serviço do posto prestado
281	Coleta hodômetro do veículo
282	Coleta a placa do veículo
283	Coleta o valor de dedução
284	Coleta o valor de acréscimo
285	Crítica o valor da transação
286	Mostra o valor
287	Mostra o vencimento
288	Coleta o valor para o Bradesco
289	Decide entre DARF e GPS
290	Decide o tipo de DARF
291	Coleta o código de receita
292	Coleta o número de referência
293	Coleta o valor juros
294	Coleta o CPF do portador
295	Exibe a mensagem de erro de valor do Bradesco
296	Coleta o CNPJ
297	Coleta o percentual
298	Coleta o mês de competência do GPS
299	Coleta o número do identificador
300	Coleta o valor de INSS
301	Coleta a receita bruta
302	Coleta a opção do operador, para confirmar, alterar e cancelar.
303	Coleta o número do Voucher
304	Coleta o valor do saque no PIN-pad
305	Scope valida o valor do saque
306	Automação valida o valor do saque
307	Confirma valor do saque no PIN-pad
308	Saque PIN-pad em andamento
314	Coleta DDD + Número de Telefone no PIN Pad
315	Redigita DDD + Número de Telefone no PIN Pad
316	Coleta dados ECF
317	Verifica se o cartão Ticket Car é benefício ou gestão de frotas
318	Verifica se há transação off-line gravada no chip do cartão Ticket Car
319	Monta a transação off-line gravada no chip do cartão Ticket Car
320	Apaga a transação off-line gravada no chip do cartão Ticket Car

339	Verifica se é débito a vista ou pagamento de carnê para GetnetLAC.
340	Coleta dados cartão presente
357	Coleta da Lista de Atualização de Preços de Mercadorias
358	Coleta da Lista de Mercadorias Consumidas
359	Decide se deve ser coletada Lista de Mercadorias
360	Decide se devem ser coletados Dados do Veículo
361	Decide se devem ser coletados Dados do Motorista
362	Decide se deve ser coletado Hodômetro do Veículo
363	Decide se deve ser coletado Horímetro do Veículo
364	Coleta Horímetro do Veículo
365	Decide se deve ser coletada a Placa do Veículo
366	Coleta de Cartão Magnético
368	Coleta de Segmento (SAV)
369	Coleta de Fornecedor (SAV)
370	Coleta de Produto/ Serviço/ Código EAN (SAV)
371	Coleta de Quantidade
372	Recupera Dados
373	Coleta de Cliente Preferencial
374	Exibe Mensagem

Código das redes

Abaixo se encontram as redes suportadas pelo SCOPE e seus respectivos códigos.

Códigos Rede Decimal	Significado	Código SAT Decimal
0	SCOPE	999
1	Tecban	31
2	Itaú	999
3	Visanet – especificação 2000.01	999
4	Bradesco	999
5	Redecard	25
6	Fininvest	999
7	Serasa	999
8	Teledata	999
9	Banrisul	999
10	Ticket	32
11	Associação Comercial de São Paulo - ACSP	999
12	BrasilCard (Antiga CNS)	999
13	Sysdata	999
14	Redecard – especificação L0102	25
15	Visanet – especificação 1998.10	999
16	CBD	999
17	Lojista	999
18	CSU	999
19	Parati	999
20	Bem	999
21	Tokoró	999
22	MaxiCred	999
23	Zogbi	999
24	ACC Card	999
25	Sorocred	30
26	Coopercred	999
27	Telesp	999
28	Policard	23
29	Via Varejo	999

Códigos Rede Decimal	Significado	Código SAT Decimal
30	SAB	999
31	E-Pharma	999
32	Vidalink	999
33	PrevSaúde	999
34	Hipercard	19
35	Interchange	999
36	Tecban – especificação 2.0	31
37	Nutricash	999
38	Losango	999
39	GoodCard	999
40	Cetelem	11
41	Bônus	999
42	American Express (AMEX) – especificação 01	3
43	Banco do Brasil GCB	999
44	Sonae	999
45	Incomm (reutilização do código da rede BankBoston)	999
46	Portal Card	999
47	Valecard	999
48	Telenet	999
49	Evangélico	999
50	Funcional Card	18
51	Ediguay	999
52	CheckCheck	999
53	Banktec	999
54	Big	999
55	Big Card	8
56	SuperCard	999
57	Banese	999
58	TR Centre	999
59	TRN Card	999
60	Infocards	999
61	Valecash	999
62	Premium	999
63	CredSystem	999
64	Redecard – especificação L02.05	25
65	E-Capture	999
66	Check Express	999
67	Conductor	999
68	ChequePre	999
69	Visanet 4.1	999
70	American Express – especificação 03.00	3
71	ECX Card	14
72	Ultragaz	999
73	GetNet	999
74	CentralCard	999
75	Orbitall	999
76	Recarga de Celular – especificação Itautec	999
77	IBI	999
78	DATASUS	999
79	Comprocard	999
80	Framaseg	999
81	Intellisys	999
82	Somar	999
83	Solucard	999
84	Oboé	999
85	Da Casa	999

Códigos Rede Decimal	Significado	Código SAT Decimal
86	Valeshop	999
87	Fidelize	999
88	UtilCard	999
89	RV Tec	999
90	GW Cel	999
91	U-Paid	999
92	Redecard – especificação L0401	25
93	Banpará	999
94	Neus	999
95	CredShop	999
96	Banco HSBC	999
97	Banco do Brasil ISO-GCB	999
98	Pharma Link	999
99	Banrisul EMV	999
100	Banestes	999
102	Cielo	12
103	Redecard – especificação L05.00	25
104	Diamante	999
105	RedeSoftnex	999
109	Usecred	999
112	DMCARD	999
113	Siscred	999
114	Epay	999
116	Orgcard	999
117	SAVS	999
118	TENDENCIA	999
121	RVTECNOLOGIA	999
123	GIVEX	999
124	CREDITEM	999
125	FOXWIN	999
126	BIN	999
127	TOPCARD	999
128	ELAVON	999
129	VR	31
133	PBM PADRAO	999
134	CONDUCTOR-PL – especificação V02.05	999

Código de especificação das redes

Abaixo se encontram as redes que possuem mais que uma especificação suportada pelo SCOPE.

Código Rede	Código Especificação	Significado	
13	1	SYSDATA	V1.XX
	2		V2.00
66	1	CHECKEXPRESS	V1.XX
	2		V2.00
73	1	GETNET-6.0	V01.00
	2		V02.00
77	1	IBI	V1.XX
	2		XSAB 1.28a
89	2	RVTEC	V2.00
90	1	GWCEL	V0003
	2		V0005
91	1	UPAID	V1.XX
	2		V2.00
102	0	CIELO	4.1.0
	1		4.1R2014
103	1	REDECARD	V5.00
	2		V5.01
	3		V5.02
118	1	TENDENCIA	V1.XX
	2		V2.00

Código das bandeiras

Como algumas transações exigem, o operador deve escolher uma bandeira e passar para o SCOPE o seu código cuja relação se encontra abaixo. Para que o operador possa escolher, é necessário que ele tenha uma lista com os códigos de cada bandeira. Uma opção é ter uma lista impressa disponível para o operador. No entanto, como geralmente nem todas as bandeiras são utilizadas para uma determinada loja, pode ser viável a aplicação, por meio de configuração, exibir na tela uma lista de opções, somente com as bandeiras utilizadas, no momento que for necessário.

Códigos Bandeira		Significado
Hexadecimal	Decimal	
0x0000	000	SCOPE
0x0001	001	Visa
0x0002	002	Mastercard
0x0003	003	Amex
0x0004	004	Lojista
0x0005	005	Dinners
0x0006	006	Sollo
0x0007	007	Cheque Eletrônico
0x0008	008	Mastercard Maestro (conhecido também como Redeshop)
0x0009	009	Itaú

Códigos Bandeira		Significado
Hexadecimal	Decimal	
0x000A	010	Bradesco
0x000B	011	Fidelidade
0x000C	012	Serasa
0x000D	013	Telecheque
0x000E	014	Sodexo Alimentacao
0x000F	015	BrasilCard (antiga RVA)
0x0010	016	Ticket Alimentação
0x0011	017	Hipercard
0x0012	018	CardCo (antiga CNS)
0x0013	019	MaxiCred
0x0014	020	Banrisul
0x0015	021	Visa Electron
0x0016	022	Redecard (bandeira para operações internas, independentes de cartão. Exemplo: Resumo de Vendas)
0x0017	023	JCB
0x0018	024	Quality Card
0x0019	025	Unnisa
0x001A	026	Fininvest
0x001B	027	Multi-Cheque
0x001C	028	VR (cartão voucher da CSU)
0x001D	029	TransCheck
0x001E	030	TecBan (bandeira genérica para serviços TECBAN)
0x001F	031	ACC Card
0x0020	032	Sorocred
0x0021	033	Parati
0x0022	034	BEM - Banco do Estado do Maranhão
0x0023	035	Tokoro
0x0024	036	Zogbi
0x0025	037	TopPremium
0x0026	038	Carrefour (cartão voucher da CooperCred)
0x0027	039	Rainbow (cartão voucher da TecBan)
0x0028	040	Telesp Celular
0x0029	041	Policard
0x002A	042	Via Financeira
0x002B	043	IBI
0x002C	044	e-Pharma (PBM)
0x002D	045	Vidalink (PBM)
0x002E	046	PrevSaude (PBM)
0x002F	047	e-Pharma balcão (solicitação de autorização via conexão direta com ScopeGW.)
0x0030	048	Aura
0x0031	049	Correspondente bancário – Unibanco
0x0032	050	Correspondente bancário – Citibank
0x0033	051	Recebimento Fininvest
0x0034	052	Multi-Benefícios
0x0035	053	VA Eletrônico (cartão alimentação)
0x0036	054	Valetik
0x0037	055	Losango
0x0038	056	Goodcard
0x0039	057	FIC
0x003A	058	Banestik (cartão voucher da TecBan)
0x003B	059	Cabal
0x003C	060	TMS
0x003D	061	Bonus (cartão voucher)
0x003E	062	Visa Vale

Códigos Bandeira		Significado
Hexadecimal	Decimal	
0x003F	063	Correspondente bancário – Banco do Brasil
0x0040	064	Colaborador (cartão de crédito do SONAE)
0x0041	065	Incomm (reutilização do código da bandeira CB-BankBoston)
0x0042	066	Portal Card
0x0043	067	Vale Card
0x0044	068	Personal Card
0x0045	069	Green Card (não mais utilizada)
0x0046	070	AsCard
0x0047	071	Evangelico
0x0048	072	Funcional card (PBM)
0x0049	073	ACSP - Associação Comercial de São Paulo
0x004A	074	ExtraBom/ABN
0x004B	075	Ediguay
0x004C	076	GoodMed (PBM)
0x004D	077	Bônus Eletrônico (cartão voucher do SONAE)
0x004E	078	RefeiSul
0x004F	079	Multi-Alimentação
0x0050	080	Multi-Cheque (Novo)
0x0051	081	CheckCheck
0x0052	082	EcxCard
0x0053	083	BigCard
0x0054	084	SuperCard
0x0055	085	Banese - Banco do Estado de Sergipe
0x0056	086	Novartis (PBM)
0x0057	087	FlexMed (PBM)
0x0058	088	TrnCentre - Transaction Centre
0x0059	089	InfoCards
0x005A	090	BaseCard
0x005B	091	SysData
0x005C	092	Correspondente bancário – Bradesco
0x005D	093	Ticket Restaurante
0x005E	094	ValeCash
0x005F	095	BankCard
0x0060	096	MedCheque
0x0061	097	Premium (cartão de crédito do SONAE)
0x0062	098	Vale Gás
0x0063	099	Datasus (PBM)
0x0064	100	BankTec
0x0065	101	PoupCard
0x0066	102	ChequePre
0x0067	103	Banquet (cartão voucher alimentação)
0x0068	104	Private Label da Redecard
0x0069	105	Cielo
0x006A	106	Rancho Convênio - PortalCard
0x006B	107	Rancho Alimentação - PortalCard
0x006C	108	PlanVale
0x006D	109	IBICard
0x006E	110	IBI PL,
0x006F	111	Pague-Conta Visanet
0x0070	112	Cartão Fácil (EPA)
0x0071	113	Cartão Fácil Losango (EPA)
0x0072	114	CPF - IBI PL deposito CDB
0x0073	115	ComproCard
0x0074	116	FarmaSeg (PBM)
0x0075	117	Unik

Códigos Bandeira		Significado
Hexadecimal	Decimal	
0x0076	118	Verde Card
0x0077	119	Rossi
0x0078	120	Somar
0x0079	121	Solucard
0x007A	122	Banquet Smart
0x007B	123	VEGAS CARD SENHA
0x007C	124	Planvale Redecard
0x007D	125	Oboé Card
0x007E	126	DaCasa
0x007F	127	CABAL Vale (TECBAN e GETNET 6.0)
0x0080	128	CABAL Débito (TECBAN e GETNET 6.0)
0x0081	129	BRTelecom Telefonia FIXA
0x0082	130	Brasil Telecom
0x0083	131	GoodVale (GETNET 6.0 Voucher)
0x0084	132	RedeSoftnex – Com senha
0x0085	133	PratiCard
0x0086	134	PrestaServ
0x0087	135	Safra Amanco
0x0088	136	SimCred
0x0089	137	Premiação SONAE
0x008A	138	Presente SONAE
0x008B	139	ValeShop
0x008C	140	PREZUNIC
0x008D	141	FAI/Orbital
0x008E	142	ConvCard
0x008F	143	Operadora Claro
0x0090	144	Operadora OI
0x0091	145	Operadora TIM
0x0092	146	Operadora Telemig
0x0093	147	Operadora Amazônia Celular
0x0094	148	Operadora Embratel
0x0095	149	Operadora Telefonica
0x0096	150	Operadora TELEMAR
0x0097	151	Operadora CTBC-Celular
0x0098	152	Operadora CTBC Fixo
0x0099	153	Operadora SERCOMTEL Celular
0x009A	154	Operadora SERCOMTEL Fixo
0x009B	155	Operadora Telefonica Família
0x009C	156	Operadora NEXTEL
0x009D	157	Operadora VIVO
0x009E	158	Cartão Parcele Mais (REDECARD L0401)
0x009F	159	BANPARA
0x00A0	160	NEUS
0x00A1	161	BANCRED SENHA
0x00A2	162	Credi-Shop
0x00A3	163	Sapore
0x00A4	164	CB-HSBC
0x00A5	165	Verocheque
0x00A6	166	Getnet
0x00A7	167	Pharmalink PBM
0x00A8	168	Aura FNAC
0x00A9	169	Aura BERGAMAIS
0x00AA	170	Com Você
0x00AB	171	BANESCARD
0x00AC	172	Nokia

Códigos Bandeira		Significado
Hexadecimal	Decimal	
0x00AD	173	Fala Fácil (Nexus)
0x00B0	176	BRASIL CONVENIOS
0x00B1	177	AMCARD
0x00B2	178	Rede MED
0x00B3	179	GPA
0x00B4	180	PAT – sem senha
0x00B5	181	PAT – com senha
0x00B6	182	FAN CARD
0x00B7	183	BANRICOMPRAS
0x00B9	185	Banrisul - Refeisul Alimentação / Refeição
0x00BA	186	Banrisul - Refeisul Combustível
0x00BB	187	TIM ON-LINE
0x00BC	188	BNB Clube
0x00BD	189	NEUS Senha
0x00BE	190	Diamante
0x00BF	191	MinasCred
0x00C0	192	Nutricash
0x00C2	194	ELO Debito
0x00C3	195	ELO Credito
0x00C4	196	Goodcard Senha
0x00C7	199	USECRED sem senha
0x00C8	200	USECRED com senha
0x00C9	201	SOROCRED Crédito
0x00CA	202	SICREDI Crédito
0x00CB	203	SICREDI Débito
0x00CC	204	BrasilCard – com senha
0x00CF	207	PL GETNET VISA
0x00D0	208	DMCARD
0x00D1	209	PLANVALE com senha
0x00D2	210	SISCRED sem senha
0x00D3	211	SISCRED com senha
0x00D4	212	DMCARD com senha
0x00D5	213	RedeSoftnex – Convênio
0x00D6	214	RedeSoftnex
0x00D7	215	Claro Off-line
0x00D8	216	Oi Fixa
0x00D9	217	Oi Off-line
0x00DA	218	Vivo off-line
0x00DB	219	Telesp Super 15
0x00DC	220	Embratel Livre Online
0x00DD	221	Epay
0x00E0	224	Orgcard débito
0x00E1	225	Orgcard crédito
0x00E2	226	Orgcard crédito DV,CS
0x00E4	228	Ticket Car
0x00E5	229	PL GETNET MASTER
0x00E6	230	SAVS
0x00E9	233	POLICARD SENHA
0X00F1	241	GIVEX
0x00F2	242	CREDITEM SENHA
0x00F4	244	MURY
0x00F5	245	ABRAPETITE
0x00F8	248	Sodexo Refeicao
0x00FA	250	TOPCARD com senha
0x00FB	251	TOPCARD sem senha

Códigos Bandeira		Significado
Hexadecimal	Decimal	
0x00FC	252	Elavon
0x00FD	253	VR ALIMENTACAO
0x00FE	254	VR AUTO
0x00FF	255	VR CULTURA
0x0100	256	VR REFEICAO
0x0101	257	MULTIBENEFICIO
0x0102	258	MULTIALIMENTACAO BEN
0x0103	259	MULTICESTABASICA
0x0104	260	CARTAO MAMAE
0x0105	261	MULTICASH
0x0106	262	CARTAO BRINQUEDO
0x0107	263	CARTAO NATAL
0x0108	264	MULTICHEQUE BEN
0x0109	265	MULTIEMPRESARIAL
0x010A	266	MULTICOMBUSTIVEL
0x010B	267	MULTICULTURA
0x010C	268	MULTIFARMA
0x010D	269	MULTIREFEICAO
0x0112	274	FORTBRASIL
0x0116	278	ELO AUTO
0x0125	293	BAHAMAS CRÉDITO
0x0126	294	BAHAMAS ALIMENTAÇÃO

Dados disponíveis das transações

Abaixo está a relação dos campos disponíveis que podem ser obtidos das transações pelas funções ScopeObtemCampoExt() e ScopeObtemCampoExt2().

CAMPO DE DADOS		VALOR DO BIT
Máscara 1	PAN – Personal Account Number (número do cartão)	0x00000001
	Valor da transação	0x00000002
	NSU (Número Sequencial Único) da transação	0x00000004
	Hora local da transação	0x00000008
	Data local da transação	0x00000010
	Data de validade do cartão	0x00000020
	Data contábil da transação	0x00000040
	Número do cheque	0x00000080
	Código de autorização	0x00000100
	Código de resposta	0x00000200
	Identificação do terminal	0x00000400
	Código do estabelecimento (contrato)	0x00000800
	Número de parcelas	0x00001000
	Taxa de serviço (gorjeta)	0x00002000
	NSU do Host	0x00004000
	Número do banco	0x00008000
	Número da agência	0x00010000
	Data de agendamento	0x00020000
	Código da bandeira	0x00040000
	Código do Serviço (*)	0x00080000
	Conteúdo do BIT 62	0x00100000
	Número do controle	0x00200000
	Código de rede	0x00400000

	Nome da bandeira	0x00800000
	Nome da Rede	0x01000000
	Trilha 02 do cartão	0x02000000
	Número de notas promissórias	0x04000000
	Código de estabelecimento Visanet	0x08000000
	Código CMC7	0x10000000
	CGC do convênio PBMS	0x20000000
	Mensagem de autenticação do cheque	0x40000000
	Saldo disponível (cartão convênio)	0x80000000
Máscara 2	NSU da transação original	0x00000001
	Cliente aderente ao seguro (IBICred)	0x00000002
	Dados do parcelado da rede Cetelem	0x00000004
	Data do movimento (Interchange, BBGCB)	0x00000008
	Nome do cedente ou empresa de convênio (Interchange, BBGCB)	0x00000010
	Lista das formas de pagamento em TEF permitidas (Interchange)	0x00000020
	Linha de autenticação (Interchange – Fininvest)	0x00000040
	Dados da consulta de fatura (Interchange – Fininvest) (Vide *ScopeApi.h*)	0x00000080
	Formas de financiamento (A: Administradora – E: Estabelecimento)	0x00000100
	Código específico da consulta AVS	0x00000200
	Pontos adquiridos ou resgatados	0x00000400
	Fator de compra	0x00000800
	NSU do Host da transação original (estornada)	0x00001000
	Identificação do cliente PBM junto à autorizadora (apenas Vidalink)	0x00002000
	Código da operadora de celular	0x00004000
	Código de área (DDD)	0x00008000
	Número do telefone	0x00010000
	ULTRAGAZ: dados do ValeGás	0x00020000
	Código IF (Instituição Financeira)	0x00040000
	Número do item da Fininvest ou Cetelem, ou número do contrato (CPCHEQUE/INSS) do IBI	0x00080000
	Valor da taxa de embarque	0x00100000
	Uso exclusivo do SONAE	0x00200000
	Informação contida no bit 124 - CDC Orbitall	0x00400000
	Código de serviço da transação original (estorno)	0x00800000
	Código de barras	0x01000000
	Permite desfazimento	0x02000000
	Logo do PAN	0x04000000
	Código da Empresa	0x08000000
	Código de Autenticação	0x10000000
	Dados do pagamento	0x20000000
	UsoRes_63	0x40000000
	Número do PDV	0x80000000
Máscara 3	Informações sobre a quantidade e os e-cupons disponíveis ao cliente	0x00000001
	Informação do desconto do resgate monetário	0x00000002
	Informações sobre a coleta de dados realizada na transação (BIT 48).	0x00000004
	Modo de Entrada (Entry Mode)	0x00000008
	Valor do Saque	0x00000010
	Resposta da consulta Infocards (bit 62 da 0110)	0x00000020

	Dados da resposta de Consulta da EPAY. Os dados retornados consistem em 3 valores de concatenados: 1. Valor Mínimo (12 dígitos) 2. Valor Máximo (12 dígitos) 3. Saldo Disponível (12 dígitos)	0x00000040
	Maximo de mercadorias permitidas para uma transacao TicketCar O dado retornado é um campo de 2 digitos.	0x00000100
	Código SAT (ver códigos em Código das redes)	0x00000200
	Versão corrente de Carga de Tabelas do Host Formato: 10 dígitos (Preenchido com zeros a esquerda, caso necessário). Disponível em transações com as seguintes Redes: • SAVS	0x00000400
	CNPJ da rede credenciadora - SAT	0x00002000

Grupo de Serviços

Nesta tabela encontram-se os grupos de serviços que o SCOPE trabalha.

Códigos Bandeira		Significado
Hexadecimal	Decimal	
0x0001	001	Cartão de débito
0x0002	002	Cartão de crédito
0x0003	003	Consulta a cheques
0x0004	004	Controle SCOPE/Autorizador
0x0005	005	Controle API/Server
0x0006	006	PAT
0x0007	007	CDC
0x0008	008	Garantia de desconto de cheques
0x0009	009	Resumo de vendas
0x000A	010	IATA – para companhias aéreas
0x000C	012	Reimpressão de comprovante
0x000E	014	Fidelidade
0x000F	015	Recarga de celular
0x0010	016	Transação financeira
0x0011	017	Investimento
0x0012	018	Medicamento
0x0013	019	Pagamento de conta
0x0014	020	Técnico
0x0017	023	Administrativo

Códigos dos Serviços

Além dos grupos de serviços, o SCOPE trabalha com o conceito de serviços. Estes serviços estão na tabela abaixo.

Código de serviço	Descrição
006	Compra com cartão de débito á vista
009	Compra com cartão de crédito á vista
013	Pré-Autorização com cartão de crédito
017	Consulta de cheques – a vista

018	Consulta de cheques – pré-datados
020	Compra com cartão de débito à vista forçada
021	Compra com cartão de débito pré-datada
022	Compra com cartão de débito parcelada sem parcela à vista
023	Compra com cartão de débito parcelada – parcela à vista
024	Compra com cartão de débito parcelada – parcela à vista forçada
027	Compra com cartão de crédito parcelado pela administradora
028	Compra com cartão de crédito parcelado pelo estabelecimento
031	Cancelamento de compra de débito
032	Cancelamento de compra de crédito
034	Compra CDC (CNS)
035	Garantia de cheques
036	Desconto de cheques
037	Solicitação de resumo de vendas
039	Compra com cartão de crédito IATA
040	Compra com cartão de crédito IATA parcelado com juros
041	Compra com cartão de crédito IATA parcelado sem juros
042	Cancelamento de compra de credito IATA
043	Cancelamento compra com cartão CDC
044	Consulta planos de pagamento para cartão CDC
045	Compra com cartão CDC
047	Consulta parcelas de crédito
048	Consulta parcelas de débito
050	Compra com cartão de débito Voucher (Alimentação)
051	Cancelamento de compra com cartão de débito Voucher (Alimentação)
056	Cancelamento de garantia de cheque
058	Consulta AVS
059	Cash
060	Cancelamento de Cash
061	Confirmação de Pré-Autorização
062	Estorno de Pré-Autorização
063	Fidelidade
064	Consulta saldo de crédito
065	Consulta Cash
068	Consulta valores possíveis de recarga de celular
069	Recarga de celular
070	Consulta saldo
071	Consulta extrato resumido
072	Consulta extrato
073	Simulação de saque
074	Saque
075	Consulta saldo de investimento
076	Consulta extrato de investimento
077	Resgate avulso
078	Resgate
079	Cancelamento de saque
080	Cancelamento de resgate
081	Obtém cartão de investimento
082	Consulta medicamento
083	Compra medicamento
084	Estorno compra medicamento
085	Pagamento de conta com cartão
086	Solicitação de autorização
087	Pagamento de conta sem cartão
088	Débito Voucher parcelado
089	Consulta pagamento de conta
090	Estorno de pagamento de conta
091	Pagamento de fatura

092	Consulta Saldo Dèbito
093	Resumo de Pagamentos
094	Baixa de O.S.
095	Teste de Comunicação
096	Estatística
097	Moedeiro
098	Compra com Cartão Dinheiro
099	Estorno da compra com Cartão Dinheiro
100	Consulta Contrato
101	Saque INSS
102	Saque Cpcheque
103	Consulta Vale Gás
104	Compra de Cheque Pré-Datada
105	Depósito CDB
106	Resgate CDB
107	Estorno de Depósito CDB
108	Estorno de Resgate CDB
109	Compra com Cartão de Crédito com juros
110	Carga de Cartão Dinheiro
111	Consulta Saldo do Cartão Dinheiro
112	Estorno da Carga de Cartão Dinheiro
113	Parcele Mais
114	Estorno do Parcele Mais
115	Transação Off-line
118	Troco Surpresa (Chance Legal)
121	Consulta Resgate de Prêmios
122	Pagamento de DARF
123	Pagamento de GPS
124	Injeção de chaves
125	Débito parcelado pela administradora
126	Envio de off-line
127	Atualização de chip
128	Autorização de Voucher
133	Simulação Crediário
134	Crediário
136	Atualização de preços
142	Cancelamento de Autorização de Voucher

Convênios

As transações relacionadas às transações de PBM exigem o código das redes, as quais são listadas abaixo.

Códigos do Convênio		Significado
Hexadecimal	Decimal	
0x0001	0001	E-Pharma
0x0002	0002	Vidalink
0x0003	0003	PrevSaude
0x0004	0004	Funcional Card
0x0005	0005	GoodCard
0x0006	0006	Novartis
0x0007	0007	FlexMed
0x0008	0008	DataSUS
0x0009	0009	FarmaSeg

Códigos do Convênio		Significado
Hexadecimal	Decimal	
0x000A	0010	Pharmalink
0x000B	0011	PBM Padrão

Apêndice B – Especificação Visanet 4.1

Com a nova especificação da Visanet, denominada Visanet 4.1, as automações já integradas ao SCOPE necessitarão se readequar e se certificar perante a Visanet. Este apêndice descreve as alterações necessárias.

Adequação

Para suportar a Visanet 4.1, os PIN-Pads terão seu *firmware* atualizado. Os PIN-Pads atualizados são denominados *PIN-Pads compartilhados*. Este novo *firmware* é compatível com as versões atuais, ou seja, também suporta as funções antigas de acesso ao PIN-Pads realizada pela biblioteca PPVISA da Visanet, e que eram utilizadas pelas aplicações de PDV. No entanto, quando o SCOPE estiver configurado para trabalhar com o PIN-Pads compartilhado, o acesso ao PIN-Pads será feito pelo SCOPE e não mais pelas aplicações de PDV.

Abaixo seguem algumas funcionalidades da Visanet 4.1 que exigirão adequação por parte da aplicação de PDV:

- Transação de venda crédito à vista ou parcelada.
- Transação de venda débito à vista, parcelada e pré-datada. Esta funcionalidade também contempla o Vale Alimentação e o Vale Refeição. Assim atende as operações da Visanet como prestadora de serviços de Acquirer para os produtos Visa Vale e Valetik. Vinculado à transação de débito à vista, o produto Compre & Saque possibilita ao portador a realização de saque em dinheiro. Durante o fluxo de coleta, desde que habilitado pela rede, o SCOPE solicitará o valor do saque através do estado TC_COLETA_VALOR_SAQUE (0xFC2D).
- O cancelamento de venda pode ser feito quando ocorrerem erros na digitação (valor, data de agenda, número de parcelas etc.) ou desistência da compra por parte do cliente, sendo necessário informar os dados da transação original a ser cancelada.
- Reimpressão de comprovante. Esta função possibilita ao lojista a impressão de uma nova cópia do comprovante da última transação realizada ou alguma específica, desde que tenha sido aprovada e se encontre no log do Concentrador TEF, funcionando como solução para o caso da existência de algum problema com a impressão original.
- Pagamento de contas. Possibilidade de pagamento de contas de concessionárias com cartão Visa Electron em estabelecimentos comerciais afiliados a Visanet (rede de farmácias, supermercados e estabelecimentos de varejo). Somente serão permitidas transações de débito à vista.
- Pré-autorização de crédito. No momento que esta transação é aprovada, a administradora de crédito sensibiliza o limite do cartão, de acordo com o valor informado, como forma de reserva e garantia ao Comércio Usuário que no momento de receber efetivamente o

pagamento da despesa. Atualmente somente hotéis e locadoras são autorizados para utilização da Pré-Autorização.

- Consulta CDC. Consulta direta ao banco emissor, possibilita o acesso a taxas aplicadas no momento da intenção de compra.
- AVS. Transação utilizada para verificação do endereço de envio da fatura do portador do cartão. Estes dados são informados pelo mesmo em uma compra feita através de cartão não presente.
- Pagamento de Fatura cartões *Private Label*.

Certificação

Uma vez que a aplicação de PDV cumpra as adequações mencionadas anteriormente, ela deve ser certificada pela Visanet.

A certificação deve ser agendada com a Itautec S.A., através do e-mail apioscope@itautec.com, e consiste em:

- Pré-homologação: realizada pela Itautec;
- Homologação: realizada pela Visanet ou por empresa designada por ela.

Apêndice C – PIN-Pad Compartilhado

IMPORTANTE: Este tópico deve ser considerado apenas pela automação que utiliza a interface coleta do SCOPE.

Abaixo segue alguns aspectos para integração com SCOPE na interface coleta:

- O SCOPE será responsável por toda a interação com o *PIN-Pad Compartilhado* num processo de TEF;
- Quando configurado o *PIN-Pad Compartilhado* no ScopeCNF, o SCOPE não retornará mais para a aplicação de PDV alguns estados de coleta, pois estes estados serão executados internamente pelo SCOPE. Os estados são:
    - TC_CARTAO
    - TC_COLETA_AUT_OU_CARTAO
    - TC_SENHA
    - TC_DECISAO_CONT
- A aplicação terá a possibilidade de interromper a interação do SCOPE com o PIN-Pad. Para isto a Aplicação PDV deverá:
    - Executar a função **ScopeConfigura** sinalizando que a aplicação PDV utilizará tal recurso (ver Configurações gerais);
    - Quando configurado, o SCOPE na leitura do cartão irá devolver o código **TC_COLETA_CARTAO_EM_ANDAMENTO (0xFCFC)** e durante as demais interações com o PIN-Pad devolverá o código **TC_COLETA_EM_ANDAMENTO (0xFCFD)** através da função **ScopeStatus**. Neste momento, a aplicação de PDV poderá decidir se continua ou interrompe o processo;
    - Para continuar o processo, a aplicação de PDV deverá executar a função ScopeResumeParam() com o parâmetro **PROXIMO_ESTADO (0x00)**, e para cancelar, **CANCELAR (0x02)**.
    - Observação: O SCOPE devolverá os códigos **TC_COLETA_EM_ANDAMENTO (0xFCFD)** e **TC_COLETA_CARTAO_EM_ANDAMENTO (0xFCFC)** ao menos uma vez por segundo.
- Para a aplicação de PDV cancelar uma leitura de cartão no PIN-Pad e realizar uma transação digitada, nos casos permitidos, serão oferecidas as seguintes opções:
    - Durante a coleta do cartão, se for acionado a tecla **Cancela** no PIN-Pad;
    - E se no estado **TC_COLETA_CARTAO_EM_ANDAMENTO (0xFCFC)** for executada a função **ScopeResumeParam,** com o parâmetro **CANCELA.**
- Se ocorrer uma das opções acima, o SCOPE devolverá o estado de coleta **TC_CARTAO_DIGITADO (0xFC85)** para receber da aplicação de PDV o número do cartão digitado.

- No arquivo scope.ini poderá ser desabilitado as opções acima - digitação do cartão (ver Sessão [PPCOMP]). Opcionalmente, a aplicação de PDV também poderá utilizar a função `ScopeConfigura` para o mesmo fim.
- O SCOPE também disponibilizará a função `ScopeValidaInterfacePP` que possibilitará a aplicação de PDV validar se está utilizando a mesma interface de acesso ao PIN-Pad que o que está configurado no SCOPE.

Apêndice D – Conjunto de bibliotecas do SCOPE Client

Neste apêndice, relacionamos o conjunto de bibliotecas e arquivos que compõe o ambiente em que se localiza o SCOPE Client para cada sistema operacional e linguagem.

MS-WINDOWS®

Para a maioria dos ambientes de programação, as bibliotecas necessárias para a execução do SCOPE Client em MS-Windows® são:

- CMC7.DLL
- COMVERIFONE.DLL
- CSMSG.DLL
- ECF4000.DLL
- ECNF.DLL
- PINPAD.DLL
- PPDIOW32.DLL
- PPGERW32.DLL
- PPINGW32.DLL
- PPSLBW32.DLL
- PPVFNW32.DLL
- PPW32.DLL
- SCOPEAPI.DLL
- SCOPECLT.DLL
- SCOPECNX.DLL
- SCOPECOM.DLL
- SCOPEECF.DLL
- SCOPEISO.DLL
- SCOPELIB.DLL
- SCOPEPRF.DLL
- SCOPEREG.DLL
- SCOPETCP.DLL
- scope.ini

Linguagem Java

Aplicativos de PDV escritos em Java, além das bibliotecas citadas acima, também precisam das seguintes:

- PINPADJAVA.DLL
- PINPADJAVA.JAR
- SCOPEJAVA.DLL

- SCOPEJAVA.JAR

Linux

Os ambientes executados no sistema operacional Linux precisam das seguintes bibliotecas:

- libScopeApi.so
- libScopeCom.so
- libScopeSerial.so
- libSenha.so
- scope.ini

Linguagem Java

Como acontece em MS-Windows®, o aplicativo de PDV escrito em Java também precisa das seguintes bibliotecas:

- libPinpadJava.so
- pinpadjava.jar
- libScopeJava.so
- scopejava.jar

Apêndice E – Identificando a versão do SCOPE Client

Antes de atualizar o SCOPE Client ou reportar um problema para a Itautec S.A. muitas vezes é necessária não só a verificação da versão do SCOPE Server, mas também do SCOPE Client. Abaixo exibimos as diversas formas de verificar a versão do SCOPE Client.

Verificando no SCOPE Server

A verificação da versão do SCOPE Client no Server, cuja tela é exibida na Figura 11, é simples. Basta clicar no PDV cuja versão do SCOPE Client deseja-se consultar, e clicar no botão "**PDV: <empresa>/<filial>/<PDV>**".

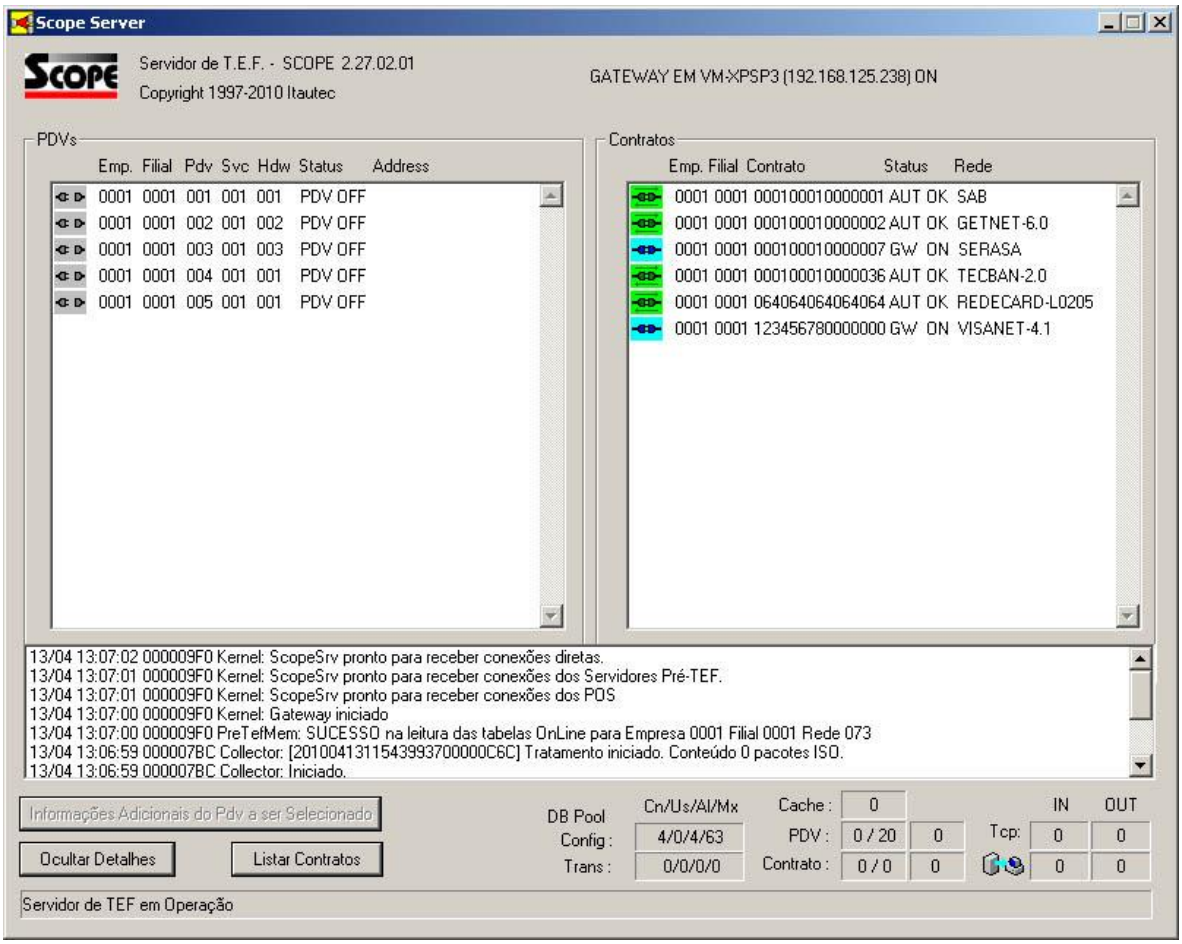


Figura 11: janela principal do servidor SCOPE

A janela similar a da Figura 12 será apresentada. Além de outras informações, nela está a versão do SCOPE Client representado no campo ScopeAPI (neste exemplo vemos 2.27.02.01). Percebe-se que embora tenha clicado num PDV específico na janela principal do servidor SCOPE, a janela com as informações do PDV possibilita a consulta de outros PDV's pelo campo "**Emp. Filial Pdv**".

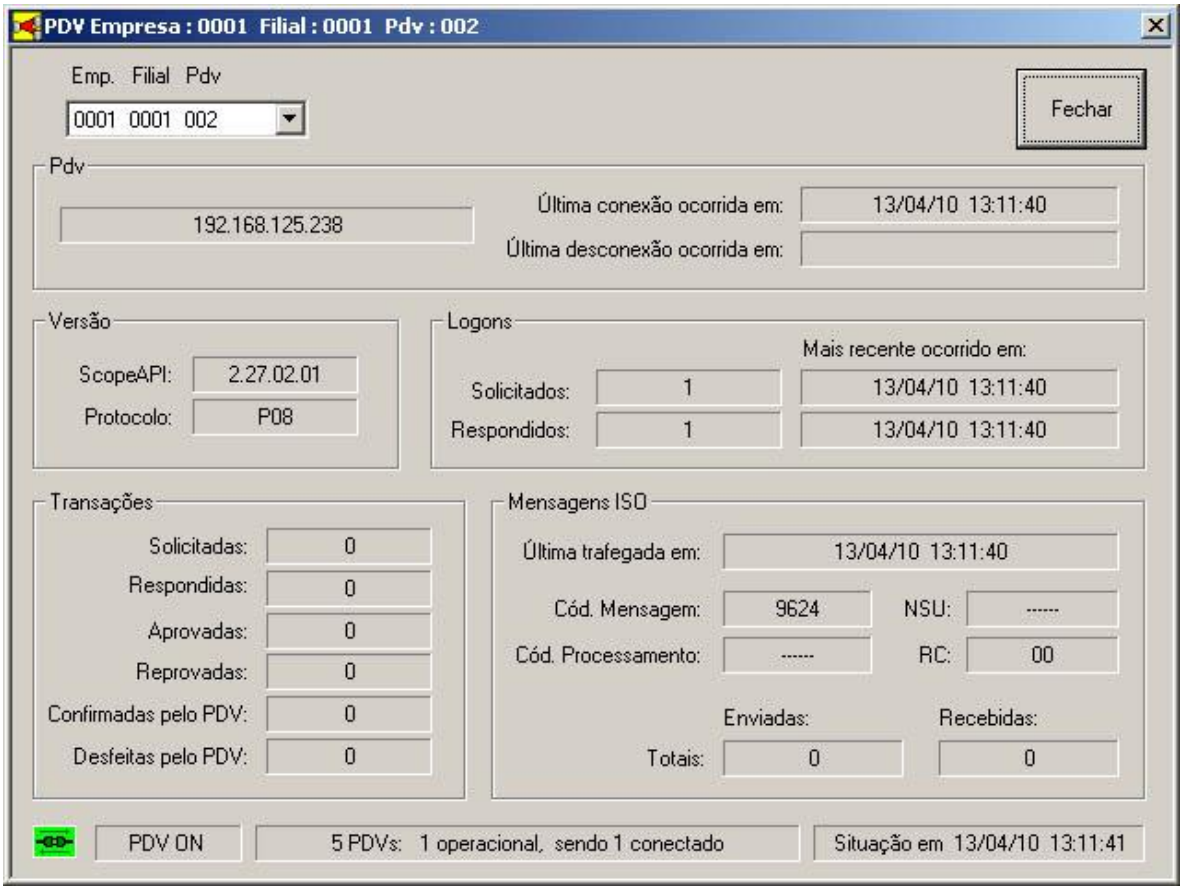


Figura 12: informações adicionais do PDV

IMPORTANTE: na consulta da versão do SCOPE Client não é preciso que este esteja conectado no servidor, mas é obrigatório que ele o tenha feito ao menos uma vez desde o *start* do servidor SCOPE.

Verificando no ambiente do PDV

Não é necessário ter acesso ao servidor SCOPE para consulta da versão do SCOPE Client e ás vezes, por política da empresa, nem todo colaborador tem acesso ao servidor. Para estes casos, deve-se verificar a versão do SCOPE Client no próprio ambiente do PDV.

SCOPE Client para MS-Windows®

Para isso, deve-se conhecer o local de instalação das DLL's do SCOPE. Localize a biblioteca ScopeAPI.dll e abra a janela de propriedades do arquivo (clicando nela com o botão direito e na opção "Propriedades"). A janela "Propriedade de ScopeAPI.dll" será aberta. Clique na aba "Versão" e será possível visualizar informações como na Figura 13. No campo "Versão do arquivo:" encontra-se a versão do SCOPE Client (no exemplo vemos "2.27.2.1", que é a mesma do exemplo da sessão anterior).

Figura 13: janela de propriedade de ScopeAPI.dll

SCOPE Client para Linux

Em Linux, no próprio nome das bibliotecas já está a versão. Elas normalmente são instaladas no diretório "`/usr/lib`" (ver Instalação do SCOPE Client para Linux), bastando a listagem dos arquivos do SCOPE. Execute:

```
$ ls libScope*
```

As bibliotecas e seus versionamentos são definidos da seguinte forma:

- libScopeApi.so.<a>.<bbb><cc>.<dd>
- libScopeCom.so.<a>.<bbb><cc>.<dd>
- libScopeJava.so.<a>.<bbb>.<e>

, onde:

- <a> é a versão da interface do SCOPE Client, cuja alteração deste implica em incompatibilidade entra as bibliotecas (exemplo: alteração de parâmetros de funções ou nomes de constantes). Este item faz parte do *soname* utilizado em Linux com *shared libraries* para prover informação de compatibilidade.
- <bbb> é a versão e o *release* do produto SCOPE. Esta versão é representada sem o ponto como, por exemplo, 225 é o SCOPE 2.25.
- <cc> é a versão do pacote do SCOPE. Perceba que não há ponto separador entre a versão, o *release* e o pacote.
- <dd> é o *build* do pacote.
- <e> é a versão da camada de interface Java.

Um exemplo da saída gerada por esse comando está na Figura 14. No exemplo, é apresentada a versão 2.25.07.01.

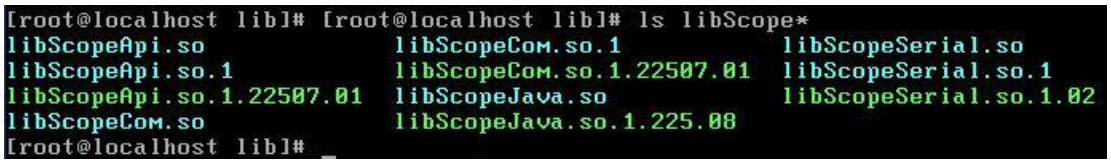


Figura 14: visualizando a versão do SCOPE Client na linha de comando do Linux

Apêndice F – Formato do Código de Barras InComm

Este apêndice tem como objetivo documentar o formato do código de barras da bandeira InComm.

O número do cartão para identificação da bandeira InComm pelo SCOPE pode ser obtido por meio de cartão magnético ou por código de barras.

O leitura e o tratamento do código de barras da InComm será realizado pela automação comercial, a qual enviará o número do PAN já formatado para o SCOPE. O layout do código de barras possui, sequencialmente, 11 dígitos refententes ao código do produto e 19 dígitos referentes ao número do cartão (PAN), o qual deve ser justificado à direita e preenchido com zeros à esquerda se for menor que 19 dígitos,como podemos observar na figura 12.

Figura 15: leiaute do código de barras da InComm

Após o código de barras ser lido e tratado pela automação, o PAN formatado será enviado para o SCOPE que irá entender o modo de entrada como cartão digitado.

Apêndice G - Glossário

A

Acquirer: veja *Adquirente*

Adquirente: é a entidade ou associação de entidades financeiras que a partir de transações efetuadas com cartões nos estabelecimentos comerciais (Merchant) associados, estabelece o devido vínculo com as entidades autorizadoras (authorizing agent) (VISA, MasterCard, Amex).

Aplicação frente de caixa: software executado no PDV com a finalidade de realizar a venda de produtos da empresa.

Aplicação de PDV: veja *Aplicação frente de caixa*.

Authorizing agent: veja *Autorizador*

Autorizador: é a organização (VISA, MasterCard, Amex) que gerencia e controla operações com cartões de crédito, passando informações entre o Adquirente e o Banco emissor.

B

Banco de dados: também conhecido como base de dados, são arquivos ou sistemas com uma estrutura regular que organizam informações. Essas estruturas podem ter a forma de uma tabela: cada tabela é composta por linhas e colunas. As informações utilizadas para um mesmo fim são agrupadas em uma base de dados.

Bandeira: entidade detentora de marcas e logotipos utilizados em cartões de crédito, débito e outros meios de pagamentos. Exemplos: Visa, Mastercard, American Express, Visa Electron, Maestro, Cheque Eletrônico.

Banco emissor: é a entidade financeira associada a uma ou mais organizações autorizadoras e que é responsável pela emissão de cartões para seus clientes.

BIN (bank identification number): número de identificação do banco representado pelos primeiros 6 dígitos do cartão.

E

Endereço IP (*Internet Protocol*): trata-se de uma tecnologia que permite a comunicação padronizada entre computadores, mesmo que estes sejam de plataformas diferentes, cada máquina possui um endereço IP que a diferencie das demais.

Estabelecimento: ou estabelecimento comercial é a entidade que aceita o cartão (card acceptor) como forma de pagamento referente à comercialização de um bem ou serviço prestado ao portador do cartão (card holder).

G

Grupo de serviço: um serviço no SCOPE pertence sempre a um grupo, que pode definir a forma de pagamento, ou ainda, a tecla finalizadora do PDV. Exemplos: cartão de crédito, cartão de débito, consulta de cheque, recarga de celular, estorno. Ver *Serviço*.

GUI: do inglês '*Graphic User Interface*' que é a interface gráfica exibida para o usuário da aplicação.

I

Issuer: Veja *Banco emissor*.

P

PDV: acrônimo para 'Ponto de Venda'. Terminal inteligente utilizado na operação de pagamento. Veja também *PoS*.

PoS: termo em inglês 'Point of Sale'. Veja também *PDV*.

R

Rede autorizadora: empresa que concentra o recebimento de transações TEF de diversos estabelecimentos, e as autoriza através de um sistema autorizador. Exemplos: Visanet, Redecard, TecBan.

S

SCOPE: O SCOPE é a solução Itautec para pagamentos eletrônicos

ScopeCNF (SCOPE Configurador): é o módulo responsável pela configuração e cadastramento de parâmetros para a solução SCOPE.

SCOPE Client: conjunto de bibliotecas localizadas na máquina em que a aplicação de frente de loja. O conjunto de bibliotecas pode variar conforme o sistema operacional e a linguagem de programação utilizada.

ScopeGW (SCOPE Gateway): funciona como um roteador de mensagens entre um ou mais servidores SCOPE e uma ou diversas redes.

ScopeSRV (SCOPE Server): é o módulo principal da solução SCOPE. É responsável por estabelecer o contato inicial de todos os contratos cadastrados na base de dados com as respectivas redes, garantindo também o fluxo de transações com estas redes

Serviço: o termo serviço no contexto do SCOPE define o objetivo de uma transação. Exemplos: compra com cartão de crédito à vista, compra com cartão de crédito parcelada pelo estabelecimento (sem juros), compra com cartão de crédito parcelada de administradora (com juros), compra com cartão de débito à vista, compra com cartão de débito pré-datado, compra com cartão de débito voucher, consulta cheque, estorno crédito, estorno débito.

Servidor: é o computador que administra e fornece programas e informações para os outros computadores conectados em rede.

SGBD: Sistema de Gerenciamento de Banco de Dados. É o conjunto de programas de computador (softwares) que controlam a criação, manutenção e uso de Bases de Dados.

T

TSR: abreviação do termo em inglês '*Terminate and Stay Resident*'. Veja Módulo residente.

© 2014

Este documento é de propriedade da **OKI Brasil Indústria e Comércio de Produtos e Tecnologia em Automação S/A**. Todos os direitos reservados.
As informações aqui contidas têm caráter técnico/informativo e não poderão ser copiadas, fotocopiadas, reproduzidas, traduzidas ou reduzidas a qualquer meio eletrônico ou forma legível por máquina sem a autorização prévia da **OKI Brasil Indústria e Comércio de Produtos e Tecnologia em Automação S/**A.
Esta se reserva, por outro lado, o direito de alterar seu conteúdo e forma, sem qualquer aviso prévio.

Produzido no Brasil.