Sicredi E-Commerce

Web Service API
Guia de integração

Versão 4

Sicredi E-commerce

Web Service API
Guia de integração

VERSÃO 4 (Brasil)

Sumário

1. Como obter suporte

Há diferentes manuais disponíveis sobre o Sicredi e-commerce. Este Guia de integração será o mais útil na integração do Web Service API para o Brasil.

Para obter informações sobre configurações, personalização, relatórios e sobre como processar transações manualmente (digitando as informações), consulte o Guia de usuário do Terminal Virtual e Portal Online.

Se você já leu a documentação e não conseguiu encontrar a resposta para sua dúvida, entre em contato conosco através da Central de Relacionamento Sicredi no 3003 4770 (capitais e regiões metropolitanas) ou 0800 7244 770 (demais localidades), ou acesse www.sicredi.com.br

2. Introdução

O Web Service API é uma interface de programação que permite a conexão da sua aplicação com o Sicredi e-commerce. Dessa forma, sua aplicação pode enviar transações de pagamento sem qualquer interferência do usuário.

Observe que, se você armazena ou processa dados de portadores de cartão na sua própria aplicação, é preciso garantir que seus componentes de sistema estejam em conformidade com as regras de segurança de dados da Indústria de cartões de pagamento (PCI DSS). Dependendo do seu volume de transações, talvez seja obrigatória uma avaliação feita por um Avaliador de Segurança Qualificado para declarar seu status de conformidade.

Do ponto de vista técnico, esta API é um web service que oferece uma operação remota para realizar transações. As três principais vantagens deste design são:

- **Independência de plataformas:** a comunicação com o Web Service API significa que sua aplicação somente precisa ser capaz de enviar e receber mensagens de SOAP. Não há requisitos atrelados a uma plataforma específica, já que a tecnologia Web Service usa um conjunto de padrões abertos. Em resumo, você pode escolher qualquer tecnologia que deseja (por ex., J2EE, .NET, PHP, ASP, etc.) para que sua aplicação possa comunicar-se com o Web Service API.
- **Integração fácil:** a comunicação com o Web Service é simples: sua aplicação precisa criar uma mensagem de solicitação SOAP para codificar sua transação, enviá-la via HTTPS para o Web Service e aguardar uma mensagem de resposta SOAP, que contém o reporte do status da sua transação. Como SOAP e HTTP foram desenhados para ser protocolos leves, a criação de solicitações e respostas é uma tarefa objetiva. Além disso, é raro precisar fazer isso manualmente, já que há inúmeras bibliotecas disponíveis em quase todas tecnologias. Em geral, a criação de uma solicitação de SOAP e a administração da resposta é reduzida a algumas linhas de código.
- **Segurança:** toda comunicação entre sua aplicação e o Web Service API tem criptografia SSL. Isso é estabelecido por sua aplicação via um certificado de cliente, que é sua identificação exclusiva no Web Service. Da mesma forma, o Sicredi e-commerce tem um certificado do servidor que sua aplicação verificará para garantir que ele esteja falando com nosso Web Service API. Por fim, seu aplicativo precisa executar uma autorização básica (nome de usuário/senha) antes de ter permissão para comunicar-se com o Web Service. Assim, os usuários que têm autorização para comunicar-se com o Sicredi e-commerce são identificados. Esses dois mecanismos de segurança garantem que os dados de transação enviados à First Data permaneçam privados e sejam identificados como dados de transação que sua aplicação informou e que não pertencem a mais ninguém.

Apesar deste breve resumo dos recursos do Web Service API, o foco deste guia é a integração da funcionalidade do Sicredi e-commerce com sua aplicação. Uma descrição detalhada, explicando como isso é feito em etapas, está presente neste guia.

3. Componentes necessários

Sustentar um alto nível de segurança exige vários componentes para a comunicação segura com o Web Service API. Como esses componentes são mencionados durante todo este guia, a lista de verificação a seguir fornecerá uma visão geral, garantindo que você recebeu o conjunto completo quando registrou sua aplicação para o Sicredi e-commerce:

- *ID da Loja (Store ID):* O ID da Loja, (por exemplo, `10012345678`), que é obrigatório para a autorização básica.
- *ID do Usuário (User ID):* o ID do Usuário indicando o usuário que tem permissão para acessar o Web Service API, por exemplo: `1`. Esse dado também é obrigatório para a autorização básica.
- *Senha:* a senha obrigatória para a autorização básica.
- *Certificado de cliente p12 (arquivo):* o certificado de cliente armazenado em um arquivo p12, com o esquema de nome `WSIDda Loja._.IDdousuário.p12 (WS`*`storeID`*`._.`*`userID`*`.p12)`, ou seja, no caso do ID da Loja/ID do Usuário acima, o arquivo seria `WS10012345678._.1.p12`. Esse arquivo é usado para autenticação do cliente no Internet Payment Gateway. Para conexão com Java, você precisará de um arquivo ks, como: `WS10012345678._.1.ks`.
- *Senha de instalação do certificado de cliente:* a senha necessária para a instalação do arquivo p12 de certificado de cliente.
- *Chave privada de certificado de cliente (Private Key) :* a chave privada do certificado de cliente armazenado em um arquivo chave, com o esquema de nome `WSIDdaLoja._.IDdousuário.key (WS`*`storeID`*`._.`*`userID`*`.key)`, ou seja, no caso do ID da Loja/ID do Usuário acima, o arquivo seria `WS10012345678._.1.key`. Algumas ferramentas que auxiliam na configuração da sua aplicaçãopara uso do Web Service API exigem esta senha durante a autenticação do certificado de cliente (client certificate) no Sicredi e-commerce
- *Senha da chave privada de certificado de cliente (client certificate):* essa senha protege a chave privada do certificado de cliente. Algumas ferramentas que auxiliam na configuração da sua aplicação para uso do Web Service API exigem esta senha durante a autenticação do certificado de cliente no Sicredi e-commerce. Ela segue o esquema de nome `ckp_`*`TimestampDeCriação (ckp_creationTimestamp)`*. Neste caso, ela seria `ckp_1193927132`.
- *Arquivo PEM do certificado de cliente:* o certificado de cliente armazenado em um arquivo PEM, com o esquema de nome `WSID da Loja._.ID do usuário.pem (WS`*`storeID`*`._.`*`userID`*`.pem)`, ou seja, no caso do ID da Loja/ID do Usuário acima, o arquivo seria `WS10012345678._.1.pem`. Algumas ferramentas que auxiliam na configuração do seu aplicativo para uso do Sicredi e-commerce exigem este arquivo em vez do arquivo p12 descrito acima.
- *Arquivo PEM do certificado do servidor:* o certificado do servidor armazenado no arquivo PEM `geotrust.pem`, que é necessário para a autenticação do servidor executando o Web Service API. Para conexão com Java, você precisará do arquivo `truststore.ks`.

Se você planeja trabalhar com venda multi-loja (vários IDs) através de sua integração, podemos emitir um Usuário API especial e um Certificado de Cliente para que você possa usar em todas as suas Lojas. Quando você envia transações desse Usuário API, você não precisa alterar o Nome do mesmo, pois o Usuário API é o mesmo para todas as suas Lojas. Você precisará incluir o ID da loja em cada pedido de transação nesse caso.

4. Como a API funciona

A seção a seguir descreve a API por meio de uma transação de cartão de crédito. O processo para outros tipos de pagamento é similar.

Na maioria dos casos, um portador de cartão inicia o processo ao comprar mercadorias ou serviços com o cartão de crédito **em sua loja online**. Em seguida, sua loja envia uma transação de cartão (principalmente para coletar os fundos do cliente) por meio do Web Service API. Depois de receber a transação, o Sicredi e-commerce a encaminha a autorização. Com base no resultado, uma aprovação ou um erro é devolvido para sua loja online. Isso significa que todos os detalhes de comunicação e processamento são cobertos pelo Sicredi e- commerce e você somente precisa sobre como comunicar-se com este Web Service.

O padrão de Web Service define esta interface pelo uso definição de linguagem de Web Service (Web Service Definition Language – WSDL)
Um arquivo WSDL que define o Web Service API para o Sicredi e-commerce pode ser encontrado em:

`https://test.ipg-online.com/ipgapi/services/order.wsdl`

Observe que você precisará fornecer seu certificado de cliente (Client Certificate), suas credenciais e o certificado do servidor (Server Certificate) ao visualizar ou solicitar o arquivo, por exemplo, em um navegador da Web. Por exemplo, caso você deseje visualizar o arquivo WSDL no Internet Explorer da Microsoft executado no Microsoft Windows XP, você primeiro deve instalar seu certificado de cliente e o certificado do servidor e, em seguida, acessar o URL acima. Isso é feito ao executar as etapas a seguir:

1. Abra a pasta na qual você salvou seu arquivo p12 do certificado de cliente.
2. Clique duas vezes no arquivo p12 do certificado de cliente.
3. Clique em *Avançar*. Verifique o nome do arquivo (que já deve estar definido no caminho do seu arquivo p12 do certificado de cliente) e clique em *Avançar*.
4. Informe a senha de instalação do certificado de cliente e clique em *Avançar*.
5. Escolha a opção *Selecionar automaticamente o armazenamento de certificados conforme o tipo de certificado* e clique em *Avançar*. Isso colocará o certificado em seu armazenamento pessoal de certificados (mais precisamente, na loja pessoal de certificados do usuário do Windows local).
6. Verifique as configurações exibidas e clique em *Terminar*. Seu certificado de cliente agora está instalado.
7. Depois, você precisa instalar o certificado do servidor. A forma mais objetiva de fazer isso é abrir a pasta na qual você salvou o arquivo PEM do certificado do servidor e renomear o arquivo para `geotrust.crt`.
8. Clique duas vezes no arquivo do certificado do servidor renomeado.
9. Clique em *Instalar Certificado*. Isso inicia o mesmo assistente.
10. Clique em *Próximo*. Selecione *Colocar todos os certificados no armazenamento a seguir* e procure pela pasta *Autoridades de Certificação Confiáveis*. Clique em *Próximo*.

11. Verifique as configurações exibidas e clique em *Concluir* (talvez seja preciso confirmar a instalação). O certificado do servidor agora está instalado no armazenamento de certificados seguros do computador local. Aqui, o Microsoft Internet Explorer pode procurar o certificado do servidor para verificação do certificado do servidor do Sicredi e-commerce recebido ao acessar a URL de WSDL acima.
12. Agora, abra a janela do Microsoft Internet Explorer e informe o URL acima no campo de endereço.
13. Depois de solicitar a URL, o servidor pedirá ao navegador para fornecer o certificado de cliente para garantir que ele esteja falando corretamente com seu aplicativo. Como você instalou o certificado nas etapas anteriores, ele é transferido para o servidor sem solicitar dados, ou seja, você não perceberá o processo. Em seguida, o Sicredi e-commerce envia seu certificado do servidor (a identificação exclusiva dele) para você. Esse certificado é verificado por meio do certificado confiável que você instalou acima. Novamente, isso é feito automaticamente sem solicitar dados. Uma conexão segura é estabelecida e todos os dados transferidos entre seu aplicativo e o Web Service API têm criptografia SSL.
14. Depois, você será direcionado para informar suas credenciais para autorização. Como nome de usuário, é preciso informar seu ID da Loja (Store ID) e ID do Usuário (User ID) codificado no formato `WS`*ID da Loja*`._.`*ID do usuário (*`WS`*storeID*`._.`*userID)*. Por exemplo, digamos que seu ID da Loja é `101`, seu ID de Usuário é `007` e sua senha é `myPW`. Você deve informar `WS101._.007` no campo de nome de usuário e `myPW` no campo de senha. Observe que suas credenciais são criptografadas antes de serem passadas para o servidor devido à conexão SSL estabelecida nas etapas anteriores. Em seguida, clique em *OK*.
15. O arquivo WSDL do Web Service API é exibido.

Em resumo, o arquivo WSDL define as operações oferecidas pelo serviço da Web, seus dados e os parâmetros de resultado, bem como essas operações podem ser chamadas. No caso do Web Service API Sicredi e-commerce, ele define apenas uma operação (`IPGApiOrder`) que pode ser chamada ao enviar a solicitação SOAP HTTP para a seguinte URL:

```
https://test.ipg-online.com/ipgapi/services
```

Os dados de entrada dessa operação é uma transação XML codificada, que devolve uma resposta XML codificada. Observe que não é necessário compreender como o arquivo WSDL é composto para uso do Sicredi e-commerce. Os capítulos a seguir são orientações para guiá-lo na configuração de sua loja para criação e realização de transações personalizadas de cartão de crédito.

Contudo, caso você esteja usando ferramentas de terceiros para ajudá-lo a configurar sua loja para acessar o Web Service API, você pode ter que fornecer a URL onde o arquivo WSDL pode ser encontrado.

De forma parecida ao descrito acima, você precisa dizer à sua ferramenta de Web Service que a comunicação está habilitada para TLS. Por isso, você precisa fornecer seu certificado de cliente e aceitar o certificado do servidor como confiável. Além disso, você deve informar suas credenciais. Como isso é feito depende muito da sua ferramenta de Web Service. Portanto, verifique a documentação da sua ferramenta para obter detalhes.

5. Como enviar transações para Sicredi e-commerce.

O objetivo deste capítulo é oferecer informações básicas sobre as etapas a serem realizadas no envio de transações para o Sicredi e-commerce. Ele descreve o que acontece se um cliente paga com seu cartão de crédito em uma loja online que utiliza o Web Service API para envio de transações.

- O cliente solicita o pagamento na loja online.
- A loja online exibe um formulário solicitando que o cliente informe os dados para o pagamento: o número de cartão de crédito, data de validade e código de segurança.
- A loja online recebe os dados e cria um documento XML codificando uma transação de *Venda* que inclui os dados informados pelo cliente e o valor total a ser pago pelo cliente.
- Depois de criar a transação de *Venda* XML, a loja online a encapsula em uma mensagem SOAP, que descreve a operação de Web Service a ser chamada com a transação XML sendo passada como um parâmetro.
- Depois de criar a mensagem SOAP, a loja online a prepara para ser transferida na Internet, incluindo seu conteúdo em uma solicitação POST de HTTPS. Além disso, a loja define os cabeçalhos de HTTP, especialmente suas credenciais (observe que as credenciais são as mesmas que você informa para visualizar o arquivo WSDL).
- A loja estabelece uma conexão TLS ao informar o certificado de cliente e de servidor.
- Em seguida, a loja online envia a requisição de HTTPS para o Web Service API e aguarda por uma resposta HTTPS.
- O Web Service API recebe a solicitação HTTPS e analisa as informações de autorização fornecidas pela loja nos cabeçalhos de HTTP.
- Com a autorização da loja para usar o Sicredi e-commerce, a mensagem SOAP contida no corpo da solicitação de HTTP é analisada. Isso ativa a operação do Web Service responsável pelo processamento da transação a ser executada.
- O Sicredi e-commerce realiza o processamento da transação, cria um documento de resposta de XML, o envolve em uma mensagem SOAP e envia essa mensagem SOAP de volta ao cliente no corpo de uma resposta de HTTPS.
- O recebimento da resposta HTTPS, “acorda” a loja que lê a mensagem SOAP e o documento XML de resposta que faz parte dela.
- Dependendo dos dados contidos no documento XML de resposta, uma página de aprovação é enviada de volta ao cliente no caso de uma transação bem-sucedida. Caso contrário, uma página de erro é apresentada.
- A página de erro ou de aprovação é exibida.

Embora este exemplo descreva o caso de uma transação de *Venda à vista (Crédito à Vista)*, outras transações seguem basicamente o mesmo processo.

Resumindo o cenário, sua aplicação precisa realizar as seguintes etapas a fim de enviar transações de cartão de crédito e analisar o resultado:

- Criar um documento XML de codificação das transações
- Envolver o documento XML em uma mensagem de solicitação SOAP

- Criar uma solicitação POST de HTTPS com as informações de identificação da sua loja informadas no cabeçalho do HTTP e da mensagem de solicitação SOAP no corpo
- Estabelecer uma conexão SSL entre sua aplicação e o Web Service API.
- Enviar a solicitação POST de HTTPS para o Sicredi e-commerce e receber a resposta
- Ler a mensagem de resposta SOAP no corpo de resposta do HTTPS
- Analisar o documento XML de resposta contido na mensagem de resposta SOAP

Essas sete etapas serão descritas nos capítulos a seguir. Eles servirão como guia durante o processo de configuração da sua aplicação para realizar transações de cartão de crédito personalizadas.

6. Como criar transações em XML

Este capítulo descreve como os diferentes tipos de transação podem ser criados em XML. Conforme descrito no cenário de exemplo acima, uma transação primeiro é codificada em um documento de XML que depois é envolvido como uma carga em uma mensagem SOAP. Isso significa que a transação com codificação XML representa o parâmetro passado para a operação do Web Service API.

Observe que há uma variedade de ferramentas de Web Service para ajudá-lo na geração de *client stubs*, o que pode evitar a necessidade de uso de XML bruto. Contudo, um conhecimento básico do formato XML é essencial para criar transações corretas, não importando o suporte de ferramenta disponível. Por isso, é recomendado familiarizar-se com o formato XML usado pelo Web Service API para codificação de transações.

6.1 Transações com cartão de crédito/débito

Independentemente do tipo de transação, a estrutura básica do documento XML de uma transação com cartão de crédito/débito é a seguinte:

```
<ipgapi:IPGApiOrderRequest
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
      <v1:Transaction>
            <v1:CreditCardTxType>...</v1:CreditCardTxType>
            <v1:CreditCardData>...</v1:CreditCardData>
            <v1:Payment>...</v1:Payment>
            <v1:TransactionDetails>...</v1:TransactionDetails>
            <v1:Billing>...</v1:Billing>
            <v1:Shipping>...</v1:Shipping>
      </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

O elemento CreditCardDataTXType é obrigatório para todas as transações com cartão de crédito. Os outros elementos dependem do tipo de transação. O conteúdo da transação é específico para o tipo. Os elementos na estrutura XML devem ser mantidos na mesma ordem mostrada nos exemplos, caso contrário, a *OrderRequest* falhará.

Para tags XML relacionadas às transações de cartão presente em uma solução de captura com um leitor de chip e senha, consulte os xsd's no apêndice deste documento.

O documento XML a seguir representa um exemplo de uma transação de *Venda à vista* usando o conjunto mínimo de elementos:

```
<ipgapi:IPGApiOrderRequest
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
      <v1:Transaction>
            <v1:CreditCardTxType>
                  <v1:Type>sale</v1:Type>
            </v1:CreditCardTxType>
            <v1:CreditCardData>
                  <v1:CardNumber>4111111111111111</v1:CardNumber>
                  <v1:ExpMonth>12</v1:ExpMonth>
                  <v1:ExpYear>07</v1:ExpYear>
            </v1:CreditCardData>
            <v1:Payment>
                  <v1:ChargeTotal>19.95</v1:ChargeTotal>
                  <v1:Currency>986</v1:Currency>
            </v1:Payment>
      </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Consulte o capítulo **Visão geral de tags de XML** para obter uma descrição detalhada de todos os elementos usados no exemplo acima, bem como elementos opcionais adicionais.

6.1.1.2 Multi-loja Venda (Sale)

O documento abaixo XML a seguir representa um exemplo de uma transação de uma venda que possui um certificado vinculado para mais de uma loja.

```
<?xml version='1.0' encoding='UTF-8'?>
  <soap-env:Envelope xmlns:soap-
env='http://schemas.xmlsoap.org/soap/envelope/'>
   <soap-env:Header />
   <soap-env:Body>
    <ipgapi:IPGApiOrderRequest xmlns:ipgapi='http://ipg-
online.com/ipgapi/schemas/ipgapi' xmlns:v1='http://ipg-
online.com/ipgapi/schemas/v1'>
     <v1:Transaction>
      <v1:CreditCardTxType>
       <v1:StoreId>2724189910</v1:StoreId>
       <v1:Type>sale</v1:Type>
      </v1:CreditCardTxType>
      <v1:CreditCardData>
       <v1:CardNumber>5547*******</v1:CardNumber>
       <v1:ExpMonth>12</v1:ExpMonth>
       <v1:ExpYear>20</v1:ExpYear>
       <v1:CardCodeValue>111</v1:CardCodeValue>
      </v1:CreditCardData>
      <v1:cardFunction>credit</v1:cardFunction>
      <v1:Payment>
       <v1:ChargeTotal>15.00</v1:ChargeTotal>
       <v1:Currency>986</v1:Currency>
      </v1:Payment>
      <v1:TransactionDetails>
       <v1:DynamicMerchantName>My Main
Website</v1:DynamicMerchantName>
      </v1:TransactionDetails>
     </v1:Transaction>
    </ipgapi:IPGApiOrderRequest>
```

```
    </soap-env:Body>
</soap-env:Envelope>
```

6.1.1.3 Multi-loja Venda (Sale) – com 3DSecure

O documento abaixo XML a seguir representa um exemplo de uma transação de uma venda que possui um certificado vinculado para mais de uma loja com 3DSecure.

```
<?xml version='1.0' encoding='UTF-8'?>
  <soap-env:Envelope xmlns:soap-
env='http://schemas.xmlsoap.org/soap/envelope/'>
    <soap-env:Header />
    <soap-env:Body>
     <ipgapi:IPGApiOrderRequest xmlns:ipgapi='http://ipg-
online.com/ipgapi/schemas/ipgapi' xmlns:v1='http://ipg-
online.com/ipgapi/schemas/v1'>
      <v1:Transaction>
       <v1:CreditCardTxType>
         <v1:StoreId>2719020033</v1:StoreId>
        <v1:Type>sale</v1:Type>
       </v1:CreditCardTxType>
       <v1:CreditCardData>
        <v1:CardNumber>5274**********</v1:CardNumber>
        <v1:ExpMonth>12</v1:ExpMonth>
        <v1:ExpYear>20</v1:ExpYear>
        <v1:CardCodeValue>785</v1:CardCodeValue>
       </v1:CreditCardData>
       <v1:CreditCard3DSecure>
        <v1:VerificationResponse>Y</v1:VerificationResponse>
        <v1:PayerAuthenticationResponse>Y</v1:PayerAuthenticationRespon
se>
        <v1:XID>qdioLX6eQnhmd3jL6vl22IdlU=</v1:XID>
        <v1:AuthenticationValue>3gomJgneJAAAAAAABwABCIJ3iUN=</v1:Authen
ticationValue>
       </v1:CreditCard3DSecure>
       <v1:cardFunction>credit</v1:cardFunction>
       <v1:Payment>
        <v1:ChargeTotal>5.00</v1:ChargeTotal>
        <v1:Currency>986</v1:Currency>
       </v1:Payment>
       <v1:TransactionDetails>
        <v1:DynamicMerchantName>My Main
Website</v1:DynamicMerchantName>
       </v1:TransactionDetails>
      </v1:Transaction>
     </ipgapi:IPGApiOrderRequest>
    </soap-env:Body>
</soap-env:Envelope>
```

6.1.2 Pré-autorização (Apenas Autorização)

O documento XML a seguir representa um exemplo de uma transação de *PreAuth* usando o conjunto mínimo de elementos:

```
<ipgapi:IPGApiOrderRequest
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
      <v1:Transaction>
            <v1:CreditCardTxType>
                  <v1:Type>preAuth</v1:Type>
            </v1:CreditCardTxType>
            <v1:CreditCardData>
                  <v1:CardNumber>4111111111111111</v1:CardNumber>
                  <v1:ExpMonth>12</v1:ExpMonth>
                  <v1:ExpYear>07</v1:ExpYear>
            </v1:CreditCardData>
            <v1:Payment>
                  <v1:ChargeTotal>100.00</v1:ChargeTotal>
                  <v1:Currency>986</v1:Currency>
            </v1:Payment>
      </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Consulte o capítulo **Visão geral de tags XML** para obter uma descrição detalhada de todos os elementos usados no exemplo acima, bem como elementos opcionais adicionais.

6.1.2.1 Captura Posterior (Post-Authorisation)

O documento XML a seguir representa um exemplo de uma transação de *Postauth* usando o conjunto mínimo de elementos. A Transação de Postauth ou Captura Posterior, é necessária para completar uma transação de Pré-autorização.

```
<ipgapi:IPGApiOrderRequest
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
      <v1:Transaction>
            <v1:CreditCardTxType>
                  <v1:Type>postAuth</v1:Type>
            </v1:CreditCardTxType>
            <v1:Payment>
                  <v1:ChargeTotal>59.00</v1:ChargeTotal>
                  <v1:Currency>986</v1:Currency>
            </v1:Payment>
            <v1:TransactionDetails>
                  <v1:OrderId>
                        703d2723-99b6-4559-8c6d-797488e8977
                  </v1:OrderId>
            </v1:TransactionDetails>
      </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Caso seu sistema não saiba o método de pagamento que foi usado para a transação original de Pré-autorização, a Captura Posterior pode ser realizada usando qualquer TxType que aceite Captura Posterior. Em seguida, o Sicredi e-commerce seleciona o método de pagamento correto com base no Número do Pedido (Order ID) de referência.

Consulte o capítulo **Visão geral de tags de XML** para obter uma descrição detalhada de todos os elementos usados no exemplo acima, bem como elementos opcionais adicionais.

6.1.2.2 ForceTicket

O documento XML a seguir representa um exemplo de uma transação de *ForceTicket* usando o conjunto mínimo de elementos:

```
<ipgapi:IPGApiOrderRequest
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
      <v1:Transaction>
            <v1:CreditCardTxType>
                  <v1:Type>forceTicket</v1:Type>
            </v1:CreditCardTxType>
            <v1:CreditCardData>
                  <v1:CardNumber>4111111111111111</v1:CardNumber>
                  <v1:ExpMonth>12</v1:ExpMonth>
                  <v1:ExpYear>07</v1:ExpYear>
            </v1:CreditCardData>
            <v1:Payment>
                  <v1:ChargeTotal>59.00</v1:ChargeTotal>
                  <v1:Currency>986</v1:Currency>
            </v1:Payment>
            <v1:TransactionDetails>
                  <v1:ReferenceNumber>123456</v1:ReferenceNumber>
            </v1:TransactionDetails>
      </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Consulte o capítulo **Visão geral de tags de XML** para obter uma descrição detalhada de todos os elementos usados no exemplo acima, bem como elementos opcionais adicionais.

6.1.2.3 Cancelamento (Return) – Cancelamento de transações a partir do dia seguinte da transação original (D+1)

O documento XML a seguir representa um exemplo de uma transação de *Cancelamento* usando o conjunto mínimo de elementos:

```
<ipgapi:IPGApiOrderRequest
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
      <v1:Transaction>
            <v1:CreditCardTxType>
                  <v1:Type>return</v1:Type>
            </v1:CreditCardTxType>
            <v1:Payment>
                  <v1:ChargeTotal>19.00</v1:ChargeTotal>
                  <v1:Currency>986</v1:Currency>
            </v1:Payment>
            <v1:TransactionDetails>
                  <v1:OrderId>
                        62e3b5df-2911-4e89-8356-1e49302b1807
                  </v1:OrderId>
            </v1:TransactionDetails>
      </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Caso seu sistema não saiba o método de pagamento que foi usado para a transação original, o Cancelamento pode ser realizado usando qualquer TxType que aceite devoluções. Em seguida, o Sicredi e-commerce seleciona o método de pagamento correto com base no Número do Pedido (Order ID) de referência.

Consulte o capítulo **Visão geral de tags de XML** para obter uma descrição detalhada de todos os elementos usados no exemplo acima, bem como elementos opcionais adicionais.

6.1.2.4 Credit Voucher

Observe que Credit Voucher é um tipo de transação que exige permissões especiais de usuário.

O documento XML a seguir representa um exemplo de uma transação de *Credit Voucher*
usando o conjunto mínimo de elementos:

```
<ipgapi:IPGApiOrderRequest
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
      <v1:Transaction>
            <v1:CreditCardTxType>
                  <v1:Type>credit</v1:Type>
            </v1:CreditCardTxType>
            <v1:CreditCardData>
                  <v1:CardNumber>4111111111111111</v1:CardNumber>
                  <v1:ExpMonth>12</v1:ExpMonth>
                  <v1:ExpYear>07</v1:ExpYear>
            </v1:CreditCardData>
            <v1:Payment>
                  <v1:ChargeTotal>50.00</v1:ChargeTotal>
                  <v1:Currency>986</v1:Currency>
            </v1:Payment>
      </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Consulte o capítulo **Visão geral de tags de XML** para obter uma descrição detalhada de todos os elementos usados no exemplo acima, bem como elementos opcionais adicionais.

6.1.2.5 Estorno (VOID) – Cancelamento de transações no mesmo dia (D0)

O documento XML a seguir representa um exemplo de uma transação de *Estorno* usando o conjunto mínimo de elementos:

```
<ipgapi:IPGApiOrderRequest
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
      <v1:Transaction>
            <v1:CreditCardTxType>
                  <v1:Type>void</v1:Type>
            </v1:CreditCardTxType>
            <v1:TransactionDetails>
                  <v1:OrderId>
                        62e3b5df-2911-4e89-8356-1e49302b1807
                  </v1:OrderId>
                  <v1:TDate>1190244932</v1:TDate>
            </v1:TransactionDetails>
      </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Caso seu sistema não saiba o método de pagamento que foi usado para a transação original, o Estorno pode ser realizado usando qualquer TxType que aceite estornos. Em seguida, o Sicredi e-commerce seleciona o método de pagamento correto com base no Numero do Pedido (Order ID) de referência e TDate. Consulte o capítulo **Visão geral de tags de XML** para obter uma descrição detalhada de todos os elementos usados no exemplo acima, bem como elementos opcionais adicionais.

7. Ações adicionais do Serviço de Web

7.1 Consultar pedido

A ação *InquiryOrder* permite obter detalhes sobre transações processadas anteriormente sobre um pedido específico. Para isso, você precisa informar o Número do Pedido (Order ID) correspondente:

```
<ns4:IPGApiActionRequest
            xmlns:ns4="http://ipg-
            online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns2:Action>
            <ns2:InquiryOrder>
                  <ns2:OrderId>
                        b5b7fb49-3310-4212-9103-5da8bd026600
                  </ns2:OrderId>
            </ns2:InquiryOrder>
      </ns2:Action>
</ns4:IPGApiActionRequest>
```

O resultado contém informações sobre todas as transações correspondentes a esse Número do Pedido (Order ID):

```
<ns4:IPGApiActionResponse
xmlns:ns4="http://ipg-
online.com/ipgapi/schemas/ipgapi"
xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns4:successfully>true</ns4:successfully>
      <ns4:OrderId>a058e6e1-fec7-4e63-a092-fab4ec689716</ns4:OrderId>
      <ns3:Billing>
            <ns3:CustomerID>customerID</ns3:CustomerID>
            <ns3:Name>äÄöÖüÜß</ns3:Name>
            <ns3:Company>company</ns3:Company>
            <ns3:Address1>address1</ns3:Address1>
            <ns3:Address2>address2</ns3:Address2>
            <ns3:City>city</ns3:City>
            <ns3:State>state</ns3:State>
            <ns3:Zip>zip</ns3:Zip>
            <ns3:Country>country</ns3:Country>
            <ns3:Phone>phone</ns3:Phone>
            <ns3:Fax>fax</ns3:Fax>
            <ns3:Email>email</ns3:Email>
      </ns3:Billing>
      <ns3:Shipping>
            <ns3:Name>shipping name</ns3:Name>
            <ns3:Address1>shipping address1</ns3:Address1>
            <ns3:Address2>shipping address2</ns3:Address2>
            <ns3:City>shipping city</ns3:City>
            <ns3:State>shipping state</ns3:State>
            <ns3:Zip>shipping zip</ns3:Zip>
            <ns3:Country>shipping country</ns3:Country>
```

```
      </ns3:Shipping>
      <ns2:TransactionValues>
            <ns3:CreditCardTxType>
                  <ns3:Type>sale</ns3:Type>
            </ns3:CreditCardTxType>
            <ns3:CreditCardData>
                  <ns3:CardNumber>5426...4979</ns3:CardNumber>
                  <ns3:ExpMonth>12</ns3:ExpMonth>
                  <ns3:ExpYear>12</ns3:ExpYear>
            </ns3:CreditCardData>
            <ns3:Payment>
                  <ns3:ChargeTotal>1</ns3:ChargeTotal>
                  <ns3:Currency>986</ns3:Currency>
            </ns3:Payment>
            <ns3:TransactionDetails>
                  <ns3:InvoiceNumber>invoice
number</ns3:InvoiceNumber>
                  <ns3:OrderId>
a058e6e1-fec7-4e63-a092-fab4ec689716
</ns3:OrderId>
                  <ns3:Ip>127.0.0.1</ns3:Ip>
                  <ns3:TDate>1273570668</ns3:TDate>
                  <ns3:TransactionOrigin>ECI</ns3:TransactionOrigin>
            </ns3:TransactionDetails>
            <ns3:Billing>
                  <ns3:CustomerID>customerID</ns3:CustomerID>
            </ns3:Billing>
            <ns4:IPGApiOrderResponse>
                  <ns4:ApprovalCode>
Y:356887:0000144820:PPXM:0612789753
</ns4:ApprovalCode>
                  <ns4:AVSResponse>PPX</ns4:AVSResponse>
                  <ns4:OrderId>
a058e6e1-fec7-4e63-a092-fab4ec689716
</ns4:OrderId>
<ns4:ProcessorApprovalCode>
356887
</ns4:ProcessorApprovalCode>
      <ns4:ProcessorReceiptNumber>
9753
</ns4:ProcessorReceiptNumber>
      <ns4:ProcessorTraceNumber>
061278
</ns4:ProcessorTraceNumber>
                  <ns4:TDate>1273570668</ns4:TDate>
                  <ns4:TDateFormatted>
2010.05.11 11:37:48 (MESZ)
</ns4:TDateFormatted>
                  <ns4:TerminalID>54000668</ns4:TerminalID>
            </ns4:IPGApiOrderResponse>
            <ns2:ReceiptNumber>9753</ns2:ReceiptNumber>
            <ns2:TraceNumber>61278</ns2:TraceNumber>
            <ns2:TransactionState>VOIDED</ns2:TransactionState>
       </ns2:TransactionValues>
      <ns2:TransactionValues>
            <ns3:CreditCardTxType>
                  <ns3:Type>credit</ns3:Type>
            </ns3:CreditCardTxType>
```

```
            <ns3:CreditCardData>
                  <ns3:CardNumber>5426...4979</ns3:CardNumber>
                  <ns3:ExpMonth>12</ns3:ExpMonth>
                  <ns3:ExpYear>12</ns3:ExpYear>
            </ns3:CreditCardData>
            <ns3:Payment>
                  <ns3:ChargeTotal>1</ns3:ChargeTotal>
                  <ns3:Currency>986</ns3:Currency>
            </ns3:Payment>
            <ns3:TransactionDetails>
            <ns3:InvoiceNumber>invoice number</ns3:InvoiceNumber>
                  <ns3:OrderId>
a058e6e1-fec7-4e63-a092-fab4ec689716
</ns3:OrderId>
                  <ns3:Ip>127.0.0.1</ns3:Ip>
                  <ns3:TDate>1273570670</ns3:TDate>
                  <ns3:TransactionOrigin>ECI</ns3:TransactionOrigin>
            </ns3:TransactionDetails>
            <ns3:Billing>
                  <ns3:CustomerID>customerID</ns3:CustomerID>
            </ns3:Billing>
            <ns4:IPGApiOrderResponse>
                  <ns4:ApprovalCode>
Y:533118:0000144821:PPXM:0160050943
</ns4:ApprovalCode>
                  <ns4:AVSResponse>PPX</ns4:AVSResponse>
                  <ns4:OrderId>
a058e6e1-fec7-4e63-a092-fab4ec689716
</ns4:OrderId>
<ns4:ProcessorApprovalCode>
533118
</ns4:ProcessorApprovalCode>
                  <ns4:ProcessorReceiptNumber>
0943
</ns4:ProcessorReceiptNumber>
                  <ns4:ProcessorTraceNumber>
016005
</ns4:ProcessorTraceNumber>
                  <ns4:TDate>1273570670</ns4:TDate>
                  <ns4:TDateFormatted>
2010.05.11 11:37:50 (MESZ)
</ns4:TDateFormatted>
                  <ns4:TerminalID>54000669</ns4:TerminalID>
            </ns4:IPGApiOrderResponse>
            <ns2:ReceiptNumber>943</ns2:ReceiptNumber>
            <ns2:TraceNumber>16005</ns2:TraceNumber>
            <ns2:TransactionState>CAPTURED</ns2:TransactionState>
      </ns2:TransactionValues>
      <ns2:TransactionValues>
            <ns3:CreditCardTxType>
                  <ns3:Type>void</ns3:Type>
            </ns3:CreditCardTxType>
            <ns3:CreditCardData>
                  <ns3:CardNumber>5426...4979</ns3:CardNumber>
                  <ns3:ExpMonth>12</ns3:ExpMonth>
                  <ns3:ExpYear>12</ns3:ExpYear>
            </ns3:CreditCardData>
            <ns3:Payment>
```

```
                <ns3:ChargeTotal>1</ns3:ChargeTotal>
                <ns3:Currency>986</ns3:Currency>
            </ns3:Payment>
            <ns3:TransactionDetails>
                <ns3:InvoiceNumber>invoice
number</ns3:InvoiceNumber>
                <ns3:OrderId>
                a058e6e1-fec7-4e63-a092-fab4ec689716
                </ns3:OrderId>
                <ns3:Ip>127.0.0.1</ns3:Ip>
                <ns3:TDate>1273570672</ns3:TDate>
                <ns3:TransactionOrigin>ECI</ns3:TransactionOrigin>
            </ns3:TransactionDetails>
            <ns3:Billing>
                <ns3:CustomerID>customerID</ns3:CustomerID>
            </ns3:Billing>
            <ns4:IPGApiOrderResponse>
                <ns4:ApprovalCode>
Y:356887:0000144820:PPX :0612799754
</ns4:ApprovalCode>
                <ns4:AVSResponse>PPX</ns4:AVSResponse>
                <ns4:OrderId>
a058e6e1-fec7-4e63-a092-fab4ec689716
</ns4:OrderId>
                <ns4:ProcessorApprovalCode>
356887
</ns4:ProcessorApprovalCode>
                <ns4:ProcessorReceiptNumber>
9754
</ns4:ProcessorReceiptNumber>
                <ns4:ProcessorTraceNumber>
061279
</ns4:ProcessorTraceNumber>
                <ns4:TDate>1273570672</ns4:TDate>
                <ns4:TDateFormatted>
2010.05.11 11:37:52 (MESZ)
</ns4:TDateFormatted>
                <ns4:TerminalID>54000668</ns4:TerminalID>
            </ns4:IPGApiOrderResponse>
            <ns2:ReceiptNumber>9754</ns2:ReceiptNumber>
            <ns2:TraceNumber>61279</ns2:TraceNumber>
        </ns2:TransactionValues>
        <ns2:TransactionValues>
            <ns3:CreditCardTxType>
                <ns3:Type>postauth</ns3:Type>
            </ns3:CreditCardTxType>
            <ns3:CreditCardData />
            <ns3:Payment>
                <ns3:ChargeTotal>1</ns3:ChargeTotal>
                <ns3:Currency>986</ns3:Currency>
            </ns3:Payment>
            <ns3:TransactionDetails>
                <ns3:OrderId>
a058e6e1-fec7-4e63-a092-fab4ec689716
                </ns3:OrderId>
                <ns3:Ip>127.0.0.1</ns3:Ip>
                <ns3:TDate>1273570673</ns3:TDate>
                <ns3:TransactionOrigin>ECI</ns3:TransactionOrigin>
```

```
            </ns3:TransactionDetails>
            <ns3:Billing>
                 <ns3:CustomerID>customerID</ns3:CustomerID>
            </ns3:Billing>
            <ns4:IPGApiOrderResponse>
            <ns4:ApprovalCode> N:-10501:invalid postauth attempt
</ns4:ApprovalCode>
                 <ns4:OrderId>
a058e6e1-fec7-4e63-a092-fab4ec689716
                 </ns4:OrderId>
                 <ns4:TDate>1273570673</ns4:TDate>
                 <ns4:TDateFormatted>
2010.05.11 11:37:53 (MESZ)
</ns4:TDateFormatted>
            </ns4:IPGApiOrderResponse>
            <ns2:TransactionState>DECLINED</ns2:TransactionState>
       </ns2:TransactionValues>
</ns4:IPGApiActionResponse>
```

7.2 Obter últimos pedidos

Esta ação apresenta uma interface de consulta para informações sobre os pedidos mais recentes que foram enviados.

7.2.1 Últimos pedidos de uma loja

Esta consulta informa “os últimos n pedidos de uma determinada loja”.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns5:IPGApiActionRequest xmlns:ns5="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/v1" xmlns:ns3="http://ipg-
online.com/ipgapi/schemas/a1"
xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
            <ns3:Action>
               <ns3:GetLastOrders>
                    <ns3:Count>5</ns3:Count>
               </ns3:GetLastOrders>
            </ns3:Action>
        </ns5:IPGApiActionRequest>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

7.2.3 Últimos pedidos de uma loja dentro de um determinado intervalo

Esta consulta informa “os últimos n pedidos de uma determinada loja *dentro de um determinado intervalo de datas”*.
Ela também pode ser usada para paginação.
As duas datas `DateFrom` e `DateTo` precisam ser informadas, no formato xs:datahora (xs:dateTime)

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
```

```
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns5:IPGApiActionRequest xmlns:ns5="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/v1" xmlns:ns3="http://ipg-
online.com/ipgapi/schemas/a1"
xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
            <ns3:Action>
                <ns3:GetLastOrders>
                    <ns3:Count>5</ns3:Count>
                    <ns3:DateFrom>2014-04-
05T10:23:37.143+02:00</ns3:DateFrom>
                    <ns3:DateTo>2014-05-
05T10:23:37.143+02:00</ns3:DateTo>
                </ns3:GetLastOrders>
            </ns3:Action>
        </ns5:IPGApiActionRequest>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

7.2.4 Todos os pedidos de uma loja após um determinado Número do Pedido (Order ID)

Esta interface foi criada para permitir a paginação de grandes conjuntos de resultados. Ela informa "Os últimos n pedidos de uma determinada loja *após um determinado pedido (por orderId)*"

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns5:IPGApiActionRequest xmlns:ns5="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/v1" xmlns:ns3="http://ipg-
online.com/ipgapi/schemas/a1"
xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
            <ns3:Action>
                <ns3:GetLastOrders>
                    <ns3:Count>2</ns3:Count>
                    <ns3:OrderID>Test SGSDAO.ConversionDate
1382020873203</ns3:OrderID>
                </ns3:GetLastOrders>
            </ns3:Action>
        </ns5:IPGApiActionRequest>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope
```

7.2.5 Resposta

Todos os métodos de consulta resultam na mesma estrutura.

- O status de sucesso é informado por `<ipgapi:successfully>true</ipgapi:successfully>`
- `<ipgapi:ResultInfo>/ <a1:MoreResultsAvailable>true</a1:MoreResultsAvailable>` avisa se há mais resultados disponíveis.
  - O serviço não tem status, portanto as consultas subsequentes de paginação precisam usar...
  - GetLastOrders(storeID, count, dateFrom, dateTo) c/ dateTo definida como a última data_pedido (order_date) do pedido do conjunto de resultados anterior OU
  - GetLastOrders(storeID, count, orderId) c/ orderId definido como o último pedido do conjunto de resultados anterior
- Lista de pedidos `<ipgapi:OrderValues>`, composta por
  - `OrderId` – identificação exclusiva do pedido (Número do Pedido)
  - `<a1:TransactionValues>` transações
  - `<v1:Basket>` a cesta
    - com itens da cesta `<v1:Item>`
    - e cada item com opções de item `<v1:Option>`

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ipgapi:IPGApiActionResponse xmlns:ipgapi="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:a1="http://ipg-
online.com/ipgapi/schemas/a1"
xmlns:pay_1_0_0="http://api.clickandbuy.com/webservices/pay_1_0_0/"
xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1">
            <ipgapi:successfully>true</ipgapi:successfully>
            <ipgapi:ResultInfo>
                <a1:MoreResultsAvailable>true</a1:MoreResultsAvailable>
            </ipgapi:ResultInfo>
            <ipgapi:OrderValues>
                <a1:OrderId>API-Test b9dfe057-7df9-4b46-bd40-9879adec3289
InquiryOrderTest::testBasket(305)</a1:OrderId>
                <a1:OrderDate>2013-10-17T14:37:06.000+02:00</a1:OrderDate>
                <v1:Basket>
                    <v1:Item>
                        <v1:ID>ID</v1:ID>
                        <v1:Description>Description</v1:Description>
                        <v1:ChargeTotal>1</v1:ChargeTotal>
                        <v1:Quantity>13</v1:Quantity>
                        <v1:Option>
                            <v1:Name>Name 2</v1:Name>
                            <v1:Choice>Choice</v1:Choice>
                        </v1:Option>
                        <v1:Option>
                            <v1:Name>Name 1</v1:Name>
                            <v1:Choice>Choice</v1:Choice>
                        </v1:Option>
                    </v1:Item>
                    <v1:Item>
                        <v1:ID>ID</v1:ID>
                        <v1:Description>Description</v1:Description>
                        <v1:ChargeTotal>1</v1:ChargeTotal>
```

```
                        <v1:Quantity>13</v1:Quantity>
                        <v1:Option>
                            <v1:Name>Name 2</v1:Name>
                            <v1:Choice>Choice</v1:Choice>
                        </v1:Option>
                        <v1:Option>
                            <v1:Name>Name 1</v1:Name>
                            <v1:Choice>Choice</v1:Choice>
                        </v1:Option>
                    </v1:Item>
                    <v1:Item>
                        <v1:ID>ID 3</v1:ID>
                        <v1:Description>Description 3</v1:Description>
                        <v1:ChargeTotal>3</v1:ChargeTotal>
                        <v1:Quantity>33</v1:Quantity>
                    </v1:Item>
                </v1:Basket>
                <v1:Billing/>
                <v1:Shipping/>
                <a1:TransactionValues>
                    <v1:CreditCardTxType>
                        <v1:Type>sale</v1:Type>
                    </v1:CreditCardTxType>
                    <v1:CreditCardData>
                        <v1:CardNumber>403587...4977</v1:CardNumber>
                        <v1:ExpMonth>12</v1:ExpMonth>
                        <v1:ExpYear>14</v1:ExpYear>
                        <v1:Brand>VISA</v1:Brand>
                    </v1:CreditCardData>
                    <v1:Payment>
                        <v1:ChargeTotal>1</v1:ChargeTotal>
                        <v1:Currency>986</v1:Currency>
                    </v1:Payment>
                    <v1:TransactionDetails>
                        <v1:OrderId>API-Test b9dfe057-7df9-4b46-bd40-
9879adec3289 InquiryOrderTest::testBasket(305)</v1:OrderId>
                        <v1:TDate>1382020626</v1:TDate>
                        <v1:TransactionOrigin>ECI</v1:TransactionOrigin>
                    </v1:TransactionDetails>
                    <ipgapi:IPGApiOrderResponse>

<ipgapi:ApprovalCode>Y:194737:0095905551:PPXM:1338513274</ipgapi:ApprovalCo
de>
                        <ipgapi:AVSResponse>PPX</ipgapi:AVSResponse>
                        <ipgapi:Brand>VISA</ipgapi:Brand>
                        <ipgapi:OrderId>API-Test b9dfe057-7df9-4b46-bd40-
9879adec3289 InquiryOrderTest::testBasket(305)</ipgapi:OrderId>
                        <ipgapi:PaymentType>CREDITCARD</ipgapi:PaymentType>

<ipgapi:ProcessorApprovalCode>194737</ipgapi:ProcessorApprovalCode>

<ipgapi:ProcessorReceiptNumber>3274</ipgapi:ProcessorReceiptNumber>

<ipgapi:ProcessorCCVResponse>M</ipgapi:ProcessorCCVResponse>

<ipgapi:ProcessorTraceNumber>133851</ipgapi:ProcessorTraceNumber>
                        <ipgapi:TDate>1382020626</ipgapi:TDate>
                        <ipgapi:TDateFormatted>2013.10.17 16:37:06
(MESZ)</ipgapi:TDateFormatted>
                        <ipgapi:TerminalID>54000668</ipgapi:TerminalID>
                    </ipgapi:IPGApiOrderResponse>
                    <a1:TransactionState>CAPTURED</a1:TransactionState>
                    <a1:UserID>1</a1:UserID>
                    <a1:SubmissionComponent>API</a1:SubmissionComponent>
```

```
            </a1:TransactionValues>
        </ipgapi:OrderValues>
        <ipgapi:OrderValues>
            <a1:OrderId>API-Test 7c59f1cf-24e3-48ae-b3fa-e7daed007564
InquiryOrderTest::testBasket(305)</a1:OrderId>
            <a1:OrderDate>2013-10-17T14:35:26.000+02:00</a1:OrderDate>
            <v1:Basket>
                <v1:Item>
                    <v1:ID>ID</v1:ID>
                    <v1:Description>Description</v1:Description>
                    <v1:ChargeTotal>1</v1:ChargeTotal>
                    <v1:Quantity>13</v1:Quantity>
                    <v1:Option>
                        <v1:Name>Name 2</v1:Name>
                        <v1:Choice>Choice</v1:Choice>
                    </v1:Option>
                    <v1:Option>
                        <v1:Name>Name 1</v1:Name>
                        <v1:Choice>Choice</v1:Choice>
                    </v1:Option>
                </v1:Item>
                <v1:Item>
                    <v1:ID>ID</v1:ID>
                    <v1:Description>Description</v1:Description>
                    <v1:ChargeTotal>1</v1:ChargeTotal>
                    <v1:Quantity>13</v1:Quantity>
                    <v1:Option>
                        <v1:Name>Name 2</v1:Name>
                        <v1:Choice>Choice</v1:Choice>
                    </v1:Option>
                    <v1:Option>
                        <v1:Name>Name 1</v1:Name>
                        <v1:Choice>Choice</v1:Choice>
                    </v1:Option>
                </v1:Item>
                <v1:Item>
                    <v1:ID>ID 3</v1:ID>
                    <v1:Description>Description 3</v1:Description>
                    <v1:ChargeTotal>3</v1:ChargeTotal>
                    <v1:Quantity>33</v1:Quantity>
                </v1:Item>
            </v1:Basket>
            <v1:Billing/>
            <v1:Shipping/>
            <a1:TransactionValues>
                <v1:CreditCardTxType>
                    <v1:Type>sale</v1:Type>
                </v1:CreditCardTxType>
                <v1:CreditCardData>
                    <v1:CardNumber>403587...4977</v1:CardNumber>
                    <v1:ExpMonth>12</v1:ExpMonth>
                    <v1:ExpYear>14</v1:ExpYear>
                    <v1:Brand>VISA</v1:Brand>
                </v1:CreditCardData>
                <v1:Payment>
                    <v1:ChargeTotal>1</v1:ChargeTotal>
                    <v1:Currency>986</v1:Currency>
                </v1:Payment>
                <v1:TransactionDetails>
                    <v1:OrderId>API-Test 7c59f1cf-24e3-48ae-b3fa-
e7daed007564 InquiryOrderTest::testBasket(305)</v1:OrderId>
                    <v1:TDate>1382020526</v1:TDate>
                    <v1:TransactionOrigin>ECI</v1:TransactionOrigin>
```

```
                    </v1:TransactionDetails>
                    <ipgapi:IPGApiOrderResponse>

<ipgapi:ApprovalCode>Y:666159:0095905550:PPXM:0656200249</ipgapi:ApprovalCo
de>
                        <ipgapi:AVSResponse>PPX</ipgapi:AVSResponse>
                        <ipgapi:Brand>VISA</ipgapi:Brand>
                        <ipgapi:OrderId>API-Test 7c59f1cf-24e3-48ae-b3fa-
e7daed007564 InquiryOrderTest::testBasket(305)</ipgapi:OrderId>
                        <ipgapi:PaymentType>CREDITCARD</ipgapi:PaymentType>

<ipgapi:ProcessorApprovalCode>666159</ipgapi:ProcessorApprovalCode>

<ipgapi:ProcessorReceiptNumber>0249</ipgapi:ProcessorReceiptNumber>

<ipgapi:ProcessorCCVResponse>M</ipgapi:ProcessorCCVResponse>

<ipgapi:ProcessorTraceNumber>065620</ipgapi:ProcessorTraceNumber>
                        <ipgapi:TDate>1382020526</ipgapi:TDate>
                        <ipgapi:TDateFormatted>2013.10.17 16:35:26
(MESZ)</ipgapi:TDateFormatted>
                        <ipgapi:TerminalID>54000669</ipgapi:TerminalID>
                    </ipgapi:IPGApiOrderResponse>
                    <a1:TransactionState>CAPTURED</a1:TransactionState>
                    <a1:UserID>1</a1:UserID>
                    <a1:SubmissionComponent>API</a1:SubmissionComponent>
                </a1:TransactionValues>
            </ipgapi:OrderValues>
        </ipgapi:IPGApiActionResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

7.3 Obter últimas transações

Esta ação fornece uma interface de consulta de informações sobre as transações mais recentes que foram enviadas.

7.3.1 Últimas transações de uma loja

Esta consulta informa “as últimas n transações de uma determinada loja”.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns5:IPGApiActionRequest xmlns:ns5="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/a1" xmlns:ns3="http://ipg-
online.com/ipgapi/schemas/v1"
xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
            <ns2:Action>
                <ns2:GetLastTransactions>
                    <ns2:count>2</ns2:count>
                </ns2:GetLastTransactions>
            </ns2:Action>
        </ns5:IPGApiActionRequest>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

7.3.2 Todas as transações de uma loja após um determinado ID da Transação

Esta interface foi criada para permitir a paginação de grandes conjuntos de resultados. Ela informa “As últimas n transações de uma determinada loja *após uma determinada transação (por transactionId {orderId, TDate})*”

Um transactionID consiste no conjunto

- `OrderId` - o ID do pedido da transação – Número do Pedido
- `TDate` - a data da transação

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns5:IPGApiActionRequest xmlns:ns5="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/a1" xmlns:ns3="http://ipg-
online.com/ipgapi/schemas/v1"
xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
            <ns2:Action>
                <ns2:GetLastTransactions>
                    <ns2:count>2</ns2:count>
                    <ns2:OrderId>A-eb65437a-c538-4cdd-82b3-
d316ae160c22</ns2:OrderId>
                    <ns2:TDate>1407373211</ns2:TDate>
                </ns2:GetLastTransactions>
            </ns2:Action>
        </ns5:IPGApiActionRequest>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

7.3.3 Resposta

Todos os métodos de consulta resultam na mesma estrutura.

7.3.3.1 O status de sucesso é retornado como

`<ipgapi:successfully>true</ipgapi:successfully>`

7.3.3.2 Listagem de transações `<a1:TransactionValues>`

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ipgapi:IPGApiActionResponse xmlns:ipgapi="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:a1="http://ipg-
online.com/ipgapi/schemas/a1"
xmlns:pay_1_0_0="http://api.clickandbuy.com/webservices/pay_1_0_0/"
xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1">
            <ipgapi:successfully>true</ipgapi:successfully>
            <a1:TransactionValues>
                <v1:CreditCardTxType>
                    <v1:Type>periodic</v1:Type>
                </v1:CreditCardTxType>
                <v1:CreditCardData>
                    <v1:CardNumber>403587...4977</v1:CardNumber>
                    <v1:ExpMonth>12</v1:ExpMonth>
                    <v1:ExpYear>14</v1:ExpYear>
                    <v1:Brand>VISA</v1:Brand>
```

```
                </v1:CreditCardData>
                <v1:Payment>
                    <v1:ChargeTotal>1</v1:ChargeTotal>
                    <v1:Currency>986</v1:Currency>
                </v1:Payment>
                <v1:TransactionDetails>
                    <v1:OrderId>A-bcbb36ad-90ad-4ff7-ad96-
b5d73dd9c5e9</v1:OrderId>
                    <v1:TDate>1407373210</v1:TDate>
                </v1:TransactionDetails>
                <ipgapi:IPGApiOrderResponse>

<ipgapi:ApprovalCode>Y:272450:0014750514:PPXM:0433836659</ipgapi:ApprovalCo
de>
                    <ipgapi:AVSResponse>PPX</ipgapi:AVSResponse>
                    <ipgapi:Brand>VISA</ipgapi:Brand>
                    <ipgapi:OrderId>A-bcbb36ad-90ad-4ff7-ad96-
b5d73dd9c5e9</ipgapi:OrderId>
                    <ipgapi:PaymentType>CREDITCARD</ipgapi:PaymentType>

<ipgapi:ProcessorApprovalCode>272450</ipgapi:ProcessorApprovalCode>

<ipgapi:ProcessorReceiptNumber>6659</ipgapi:ProcessorReceiptNumber>

<ipgapi:ProcessorCCVResponse>M</ipgapi:ProcessorCCVResponse>

<ipgapi:ProcessorTraceNumber>043383</ipgapi:ProcessorTraceNumber>

<ipgapi:ReferencedTDate>1407373210</ipgapi:ReferencedTDate>
                    <ipgapi:TDate>1407373210</ipgapi:TDate>
                    <ipgapi:TDateFormatted>2014.08.07
03:00:10 (CEST)</ipgapi:TDateFormatted>
                    <ipgapi:TerminalID>54000667</ipgapi:TerminalID>
                </ipgapi:IPGApiOrderResponse>
                <a1:TransactionState>CAPTURED</a1:TransactionState>
                <a1:UserID>1</a1:UserID>
                <a1:SubmissionComponent>BUS</a1:SubmissionComponent>
            </a1:TransactionValues>
            <a1:TransactionValues
                    >
                <v1:CreditCardTxType>
                    <v1:Type>periodic</v1:Type>
                </v1:CreditCardTxType>
                <v1:CreditCardData>
                    <v1:CardNumber>403587...4977</v1:CardNumber>
                    <v1:ExpMonth>12</v1:ExpMonth>
                    <v1:ExpYear>14</v1:ExpYear>
                    <v1:Brand>VISA</v1:Brand>
                </v1:CreditCardData>
                <v1:Payment>
                    <v1:ChargeTotal>1</v1:ChargeTotal>
                    <v1:Currency>986</v1:Currency>
                </v1:Payment>
                <v1:TransactionDetails>
                    <v1:OrderId>A-52421c39-69c4-4b2d-959d-
9fdcd3a9420a</v1:OrderId>
                    <v1:TDate>1407373209</v1:TDate>
                </v1:TransactionDetails>
                <ipgapi:IPGApiOrderResponse>

<ipgapi:ApprovalCode>Y:416502:0014750513:PPXM:4625106408</ipgapi:ApprovalC
o de>
                    <ipgapi:AVSResponse>PPX</ipgapi:AVSResponse>
                    <ipgapi:Brand>VISA</ipgapi:Brand>
```

```
                    <ipgapi:OrderId>A-52421c39-69c4-4b2d-959d-
9fdcd3a9420a</ipgapi:OrderId>
                    <ipgapi:PaymentType>CREDITCARD</ipgapi:PaymentType>

<ipgapi:ProcessorApprovalCode>416502</ipgapi:ProcessorApprovalCode>

<ipgapi:ProcessorReceiptNumber>6408</ipgapi:ProcessorReceiptNumber>

<ipgapi:ProcessorCCVResponse>M</ipgapi:ProcessorCCVResponse>

<ipgapi:ProcessorTraceNumber>462510</ipgapi:ProcessorTraceNumber>

<ipgapi:ReferencedTDate>1407373209</ipgapi:ReferencedTDate>
                    <ipgapi:TDate>1407373209</ipgapi:TDate>
                    <ipgapi:TDateFormatted>2014.08.07
03:00:09 (CEST)</ipgapi:TDateFormatted>
                    <ipgapi:TerminalID>54000666</ipgapi:TerminalID>
                </ipgapi:IPGApiOrderResponse>
                <a1:TransactionState>CAPTURED</a1:TransactionState>
                <a1:UserID>1</a1:UserID>
                <a1:SubmissionComponent>BUS</a1:SubmissionComponent>
            </a1:TransactionValues>
        </ipgapi:IPGApiActionResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

7.4 Pagamentos recorrentes

A ação *RecurringPayment* permite programar, modificar ou cancelar pagamentos periódicos.

7.4.1 Programar um pagamento recorrente (Install)

O exemplo a seguir mostra como programar um pagamento mensal de cartão de crédito com 12 ocorrências (*InstallmentCount*) em 2011, iniciando em 15 de janeiro de 2011:

```
<ns4:IPGApiActionRequest
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
     <ns2:Action>
            <ns2:RecurringPayment>
                 <ns2:Function>install</ns2:Function>
                 <ns2:RecurringPaymentInformation>
                        <ns2:RecurringStartDate>
                              20110115
                        </ns2:RecurringStartDate>
                        <ns2:InstallmentCount>12</ns2:InstallmentCount>
                        <ns2:InstallmentFrequenc
                              y> 1
                        </ns2:InstallmentFrequency>
                        <ns2:InstallmentPerio
                              d> month
                        </ns2:InstallmentPeriod>
                  </ns2:RecurringPaymentInformation>
                  <ns2:CreditCardData>
                        <ns3:CardNumber>4035875676474977</ns3:CardNumbe
                        r>
                        <ns3:ExpMonth>12</ns3:ExpMonth>
                        <ns3:ExpYear>12</ns3:ExpYear>
                        <ns3:CardCodeValue>977</ns3:CardCodeValue>
```

```
                </ns2:CreditCardData>
                <ns3:Payment>
                      <ns3:ChargeTotal>1</ns3:ChargeTotal>
                      <ns3:Currency>986</ns3:Currency>
                </ns3:Payment>
          </ns2:RecurringPayment>
     </ns2:Action>
</ns4:IPGApiActionRequest>
```

Se você definir a *RecurringStartDate* como a datal atual, o primeiro pagamento será iniciado imediatamente. Neste caso, os dados de pagamento somente serão armazenados para pagamentos futuros se o primeiro pagamento for bem-sucedido/aprovado.
Uma data de início no passado não é permitida.

O valor padrão para *TransactionOrigin* é „ECI". Se você deseja alterar esse valor, você pode enviar uma tag TransactionOrigin diferente na tag RecurringPayment.

7.4.2 Modificar (Modify)

As modificações de um Pagamento Recorrente podem ser iniciadas usando o *Número do Pedido (Order Id)*:

```
<ns4:IPGApiActionRequest
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
     <ns2:Action>
            <ns2:RecurringPayment>
                  <ns2:Function>modify</ns2:Function>
                  <ns2:OrderId>
                        e368a525-173f-4f56-9ae2-beb4023a6993
                  </ns2:OrderId>
                  <ns2:RecurringPaymentInformation>
                        <ns2:InstallmentCount>999</ns2:InstallmentCount
                        >
                  </ns2:RecurringPaymentInformation>
            </ns2:RecurringPayment>
     </ns2:Action>
</ns4:IPGApiActionRequest>
```

Somente é necessário incluir elementos que precisam ser alterados. Se você alterar o número de cartão de crédito, também será preciso incluir a data de vencimento. Caso contrário, você pode alterar a data de vencimento sem especificar o número de cartão de crédito. Se você deseja alterar o valor, também é preciso incluir a moeda.

7.4.3 Cancelar (Cancel)

Para cancelar um Pagamento Recorrente, você também usa o *Número do Pedido (Order Id)*:

```
<ns4:IPGApiActionRequest
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
     <ns2:Action>
            <ns2:RecurringPayment>
                  <ns2:Function>cancel</ns2:Function>
                  <ns2:OrderId>
                        e368a525-173f-4f56-9ae2-beb4023a6993
```

```
                </ns2:OrderId>
            </ns2:RecurringPayment>
     </ns2:Action>
</ns4:IPGApiActionRequest>
```

7.4.4 Testar Pagamentos recorrentes em um ambiente de teste

O sistema de testes permite iniciar manualmente um pagamento agendado para testar esta funcionalidade. Essa função não funciona no ambiente de Produção.

```
<ns4:IPGApiActionRequest
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
     <ns2:Action>
            <ns2:RecurringPayment>
                  <ns2:Function>
                        perform only in test environment
                  </ns2:Function>
                  <ns2:OrderId>
                        A-eab002b9-5889-4082-9cc9-5bc06b8eaa61
                  </ns2:OrderId>
            </ns2:RecurringPayment>
     </ns2:Action>
</ns4:IPGApiActionRequest>
```

7.4.5 Resposta

A resposta para uma parcela, modificação ou cancelamento bem-sucedido contém o valor ***true*** para o parâmetro *<ns4:successfully>*:

```
<ns4:IPGApiActionResponse
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns4:successfully>true</ns4:successfully>
      <ns4:OrderId>e368a525-173f-4f56-9ae2-beb4023a6993</ns4:OrderId>
</ns4:IPGApiActionResponse>
```

7.5 Status de transação externa

Alguns endpoints de pagamento não enviam o resultado final de uma transação de pagamento em sua resposta. Nesses casos, o Sicredi e-commerce devolve um código de autorização que começa com um ponto de interrogação (?:…). A ação GetExternalTransactionState permite solicitar atualizações no estado dessas transações.

7.6 Ativar e-mails de notificações

A ação SendEMailNotification ativa e-mails de notificação para uma determinada transação. O e-mail será criado com o modelo de e-mail que foi configurado para sua loja.

Consulte o Guia de usuário do Terminal Virtual e Portal online para obter mais informações sobre notificações de transações por e-mail.

```
<ns5:IPGApiActionRequest
            xmlns:ns5=http://ipg-online.com/ipgapi/schemas/ipgapi
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
     <ns2:Action>
            <ns2:SendEMailNotification>
                  <ns2:OrderId>0/8/15</ns2:OrderId>
                  <ns2:TDate>1250599046</ns2:TDate>
            </ns2:SendEMailNotification>
     </ns2:Action>
</ns5:IPGApiActionRequest>
```

Se o parâmetro opcional Email não estiver definido, o endereço de e-mail do cliente armazenado com a transação será usado.

7.7 Informações da cesta e Catálogo de produtos

7.7.1 Informações da cesta em mensagens de transação

O exemplo a seguir mostra como você pode usar os parâmetros da cesta para documentar na transação o que foi vendido.

```
<ns5:IPGApiOrderRequest
            xmlns:ns5="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
      <ns2:Transaction>
            <ns2:CreditCardTxType>
                  <ns2:Type>sale</ns2:Type>
            </ns2:CreditCardTxType>
            <ns2:CreditCardData>
                  <ns2:CardNumber>403587XXXXXX4977</ns2:CardNumber>
                  <ns2:ExpMonth>12</ns2:ExpMonth>
                  <ns2:ExpYear>14</ns2:ExpYear>
            </ns2:CreditCardData>
            <ns2:Payment>
                  <ns2:ChargeTotal>1</ns2:ChargeTotal>
                  <ns2:Currency>EUR</ns2:Currency>
            </ns2:Payment>
            <ns2:TransactionDetails>
                  <ns2:OrderId>68d4a595-fd58-4859-83cd-
1ae13962a3ac</ns2:OrderId>
            </ns2:TransactionDetails>
            <ns2:Basket>
                  <ns2:Item>
                        <ns2:ID>product ID xyz</ns2:ID>
                        <ns2:Description>description of
abc</ns2:Description>
```

```
<ns2:ChargeTotal>11</
ns2:ChargeTotal>
<ns2:Currency>EUR</ns
2:Currency>
<ns2:Quantity>5</ns2:
Quantity>
<ns2:Option>
      <ns2:Name>colou
      r</ns2:Option>
      <ns2:Choice>blu
      e</ns2:Choice>
</ns2:Option>
<ns2:Option>
      <ns2:Name>size<
      /ns2:Option>
      <ns2:Choice>lar
      ge</ns2:Choice>
</ns2:Option>
```

```
                </ns2:Item>
            </ns2:Basket>
      </ns2:Transaction>
</ns5:IPGApiOrderRequest>
```

7.7.2 Como configurar um Catálogo de produtos

Você pode armazenar informações básicas sobre os produtos que você vende da seguinte forma:

```
<ns5:IPGApiActionRequest
            xmlns:ns5="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
      <ns3:Action>
            <ns3:ManageProducts>
                  <ns3:Function>store</ns3:Function>
                  <ns3:Product>
                        <ns3:ProductID>product ID xyz</ns3:ProductID>
                        <ns2:ChargeTotal>2</ns2:ChargeTotal>
                        <ns2:Currency>EUR</ns2:Currency>
                        <ns3:OfferStarts>
                              2014-12-27T13:29:41.000+01:00
                        </ns3:OfferStarts>
                        <ns3:OfferEnds>
                              2015-09-19T14:29:41.000+02:00
                        </ns3:OfferEnds>
                        <ns2:Option>
                              <ns2:Name>colour</ns2:Option>
                              <ns2:Choice>blue</ns2:Choice>
                        </ns2:Option>
                        <ns2:Option>
                              <ns2:Name>size</ns2:Option>
                              <ns2:Choice>large</ns2:Choice>
                        </ns2:Option>
                  </ns3:Product>
            </ns3:ManageProducts>
      </ns3:Action>
</ns5:IPGApiActionRequest>
```

OfferStarts e OfferEnds são opcionais e podem ser usados para restringir a visibilidade dos produtos relacionados em aplicativos personalizados, mas esses campos não restringem a possibilidade de uma venda. Há mais campos opcionais, como Descrição, OptionName e Nome. Analise o campo a1.xsd no anexo deste documento.

A função *exibir* (display) mostra o produto selecionado com todas as suas características.

```
<ns5:IPGApiActionRequest
            xmlns:ns5="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
      <ns3:Action>
            <ns3:ManageProducts>
                  <ns3:Function>display</ns3:Function>
                  <ns3:Product>
                        <ns3:ProductID>product ID xyz</ns3:ProductID>
                  </ns3:Product>
            </ns3:ManageProducts>
      </ns3:Action>
</ns5:IPGApiActionRequest>
```

A função *excluir* (delete) pode ser usada para definir o estoque disponível de um produto

como zero.

```
<ns5:IPGApiActionRequest
xmlns:ns5="http://ipg-online.com/ipgapi/schemas/ipgapi"
xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
       <ns3:Action>
            <ns3:ManageProducts>
                  <ns3:Function>delete</ns3:Function>
                  <ns3:Product>
                        <ns3:ProductID>product ID xyz</ns3:ProductID>
                  </ns3:Product>
            </ns3:ManageProducts>
       </ns3:Action>
</ns5:IPGApiActionRequest>
```

7.7.3 Gerenciar estoque de produtos

Para cada função de estoque de produtos, deve haver o ID do produto (Product ID) e suas opções no seu Catálogo de produtos.

Depois de instalar um produto, é possível preencher o estoque de produto com a função adicionar (add).

```
<ns5:IPGApiActionRequest
xmlns:ns5=http://ipg-online.com/ipgapi/schemas/ipgapi
xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1" xmlns:ns3="http://ipg-
online.com/ipgapi/schemas/a1"
xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
       <ns3:Action>
            <ns3:ManageProductStock>
                  <ns3:Function>add</ns3:Function>
                  <ns3:ProductStock>
                        <ns3:ProductID>product ID xyz</ns3:ProductID>
                        <ns2:Option>
                              <ns2:Name>colour</ns2:Option>
                              <ns2:Choice>blue</ns2:Choice>
                        </ns2:Option>
                        <ns2:Option>
                              <ns2:Name>size</ns2:Option>
                              <ns2:Choice>large</ns2:Choice>
                        </ns2:Option>
                        <ns3:Quantity>13</ns3:Quantity>
                  </ns3:ProductStock>
            </ns3:ManageProductStock>
      </ns3:Action>
</ns5:IPGApiActionRequest>
```

A função Subtrair (Substract) funciona da mesma forma, mas somente altera a quantidade, se a diferença não é negativa. Se você deseja definir a quantidade como zero, é possível usar a função excluir descrita acima.

7.7.4 Transações de venda usando o estoque de produtos

Depois de configurar o estoque de produtos, você pode usá-lo para verificar se há itens suficientes no estoque para uma transação. Em seguida, uma transação bem-sucedida vai subtrair a quantidade. Se o estoque de produto contém menos do que a quantidade solicitada, a transação é rejeitada sem qualquer alteração no estoque do produto.

Para usar essa função, adicione **`<ns2:ProductStock>check</ns2:ProductStock>`** à Cesta.

```
<ns5:IPGApiOrderRequest
xmlns:ns5="http://ipg-online.com/ipgapi/schemas/ipgapi"
xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
      <ns2:Transaction>
            <ns2:CreditCardTxType>
                  <ns2:Type>sale</ns2:Type>
            </ns2:CreditCardTxType>
            <ns2:CreditCardData>
                  <ns2:CardNumber>403587XXXXXX4977</ns2:CardNumber>
                  <ns2:ExpMonth>12</ns2:ExpMonth>
                  <ns2:ExpYear>14</ns2:ExpYear>
            </ns2:CreditCardData>
            <ns2:Payment>
                  <ns2:ChargeTotal>1</ns2:ChargeTotal>
                  <ns2:Currency>EUR</ns2:Currency>
            </ns2:Payment>
            <ns2:TransactionDetails>
                  <ns2:OrderId>68d4a595-fd58-4859-83cd-
1ae13962a3ac</ns2:OrderId>
            </ns2:TransactionDetails>
            <ns2:Basket>
<ns2:ProductStock>check</ns2:ProductStock>
                  <ns2:Item>
                        <ns2:ID>product ID xyz</ns2:ID>
                        <ns2:Description>description of
abc</ns2:Description>
                        <ns2:ChargeTotal>11</ns2:ChargeTotal>
                        <ns2:Currency>EUR</ns2:Currency>
                        <ns2:Quantity>5</ns2:Quantity>
                        <ns2:Option>
                              <ns2:Name>colour</ns2:Option>
                              <ns2:Choice>blue</ns2:Choice>
                        </ns2:Option>
                        <ns2:Option>
                              <ns2:Name>size</ns2:Option>
                              <ns2:Choice>large</ns2:Choice>
                        </ns2:Option>
                  </ns2:Item>
            </ns2:Basket>
      </ns2:Transaction>
</ns5:IPGApiOrderRequest>
```

8. Data Vault (Tokenização)

Com o produto Data Vault, você pode armazenar dados de cartão em um banco de dados criptografado no data Center da SICREDI. Esses dados poderão ser usados posteriormente para o envio de transações, sem que você precise armazenar dados sensíveis de cartão nos seus sistemas.

Se você solicitou essa opção de produto, o Web Service API oferece as seguintes funções.

Consulte mais possibilidades com o produto do Data Vault no Guia de integração da solução Connect.

8.1 Armazenar ou atualizar as informações de pagamento ao realizar uma transação

Você pode enviar o parâmetro ***HostedDataID,*** juntamente com os dados da transação, como uma identificação exclusiva para as informações de pagamento desta transação. Dependendo do tipo de pagamento, número do cartão de crédito e data de vencimento serão armazenados neste ID (HostedDataID). Em casos onde o HostedDataID enviado já existe para sua loja, as informações de pagamento armazenadas serão atualizadas.

```
<ipgapi:IPGApiOrderRequest
xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
            <v1:Transaction>
                  <v1:CreditCardTxType>
                        <v1:Type>sale</v1:Type>
                  </v1:CreditCardTxType>
                  <v1:CreditCardData>
                        <v1:CardNumber>4111111111111111</v1:CardNumber>
                        <v1:ExpMonth>12</v1:ExpMonth>
                        <v1:ExpYear>07</v1:ExpYear>
                  </v1:CreditCardData>
                  <v1:Payment>
                        <v1:HostedDataID>
HDID customer 1234567
            </v1:HostedDataID>
                  <v1:ChargeTotal>19.00</v1:ChargeTotal>
                  <v1:Currency>986</v1:Currency>
                  </v1:Payment>
            </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

O registro somente será armazenado se a autorização da transação de pagamento for bem-sucedida e sua Loja foi configurada para esse serviço.

8.2 Armazenar informações de pagamento para uma transação aprovada

As informações de pagamento também podem ser armazenadas em relação a uma transação aprovada anteriormente.

```
        <ns4:IPGApiActionRequest
xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1" xmlns:ns3="http://ipg-
online.com/ipgapi/schemas/v1">
            <ns2:Action>
                <ns2:StoreHostedData>
                    <ns2:DataStorageItem>
                        <ns2:OrderId>1234567890</ns2:OrderId>
                        <ns2:HostedDataID>
4e72021b-d155-4062-872a-30228c0fe023
</ns2:HostedDataID>
                    </ns2:DataStorageItem>
                </ns2:StoreHostedData>
            </ns2:Action>
        </ns4:IPGApiActionRequest>
```

Esta ação armazena as informações de pagamento da transação com o número de pedido (Order ID) 1234567890. A transação deve ser uma transação aprovada; caso contrário, essa ação falhará.

8.3 Iniciar transações de pagamento usando dados armazenados

Se as informações do portador de cartão foram armazenadas com o produto Data Vault, você pode executar transações usando o HostedDataID sem a necessidade de enviar os dados do cartão de crédito novamente.
Observe que não é permitido armazenar o código de segurança do cartão (presente na maioria dos casos, no verso do cartão) de forma que o portador de cartão ainda precisa inserir esse valor para transações de cartão de crédito. Para o processo de finalização de compra na sua loja na Web, recomendamos também que você armazene os últimos quatro dígitos do número do cartão de crédito no seu sistema e os apresente no momento do pagamento. Assim, o portador de cartão pode ver qual dos seus vários cartões está armazenado na sua loja e será usado para essa transação de pagamento.

```
<ipgapi:IPGApiOrderRequest
      xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
      xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
      <v1:Transaction>
            <v1:CreditCardTxType>
                  <v1:Type>sale</v1:Type>
            </v1:CreditCardTxType>
            <v1:Payment>
                  <v1:HostedDataID>
                        HDID customer 1234567
                  </v1:HostedDataID>
                  <v1:ChargeTotal>19.00</v1:ChargeTotal>
                  <v1:Currency>986</v1:Currency>
            </v1:Payment>
      </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

8.4 Armazenar informações de pagamento sem realizar uma transação ao mesmo tempo

Além da possibilidade de armazenar novos registros ao realizar uma transação de pagamento, é possível armazenar informações de pagamento via solicitação. Dessa forma, você também pode carregar vários registros de uma vez. O exemplo a seguir mostra o upload de registros com dados de cartão de crédito e debito direto alemão (não disponível para o Brasil). Observe que, neste caso, os registros existentes serão atualizados se o *HostedDataID* for o mesmo.

```
<ns4:IPGApiActionRequest
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns2:Action>
            <ns2:StoreHostedData>
                  <ns2:DataStorageItem>
                        <ns2:CreditCardData>
                              <ns3:CardNumber>
                                    4035875676474977
                              </ns3:CardNumber>
                              <ns3:ExpMonth>12</ns3:ExpMonth>
                              <ns3:ExpYear>08</ns3:ExpYear>
                        </ns2:CreditCardData>
                        <ns2:HostedDataID>
                              d763bba7-1cfa-4d3d-94af-9fbe29ec0e26
                        </ns2:HostedDataID>
                  </ns2:DataStorageItem>
                  <ns2:DataStorageItem>
                        <ns2:DE_DirectDebitData>
                              <ns3:BankCode>50014560</ns3:BankCode>
                              <ns3:AccountNumber>
                                    32121503
                              </ns3:AccountNumber>
                        </ns2:DE_DirectDebitData>
                        <ns2:HostedDataID>
                              691c7cb3-a752-4d6d-abde-83cad63de258
                        </ns2:HostedDataID>
                  </ns2:DataStorageItem>
            </ns2:StoreHostedData>
      </ns2:Action>
</ns4:IPGApiActionRequest>
```

O resultado de um armazenamento bem-sucedido contém o valor ***true*** para o parâmetro <*ns4:successfully*>:

```
<ns4:IPGApiActionResponse
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns4:successfully>true</ns4:successfully>
</ns4:IPGApiActionResponse>
```

Em casos onde um ou mais registros não foram armazenados corretamente, os IDs de Dados Hospedados (Hosted Data IDs) correspondentes são marcados no resultado:

```
<ns4:IPGApiActionResponse
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns4:successfully>true</ns4:successfully>
      <ns2:Error Code="SGSDAS-020300">
            <ns2:ErrorMessage>
                  Could not store the hosted data id:
                  691c7cb3-a752-4d6d-abde-83cad63de258.
                  Reason: An internal error has occured while
                  processing your request
            </ns2:ErrorMessage>
      </ns2:Error>
</ns4:IPGApiActionResponse>
```

Evitar dados de portador de cartão duplicados para vários registros

Para evitar que clientes usem os mesmos dados de portador de cartão para contas de vários usuários, a tag adicional ***DeclineHostedDataDuplicates*** pode ser enviada junto com a solicitação. Os valores válidos para essa tag são “true” ou “false”. Se o valor para essa tag é definido como “true” e os dados do portador do cartão na solicitação já são associados com outro “hosteddataid”, a transação será recusada.

8.5 Exibir registros armazenados

Os registros existentes serão exibidos usando a ação *Exibir (Display)*:

```
<ns4:IPGApiActionRequest
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1">
      <ns3:Action>
            <ns3:StoreHostedData>
                  <ns3:DataStorageItem>
                        <ns3:Function>display</ns3:Function>
                        <ns3:HostedDataID>
                              d56feaaf-2d96-4159-8fd6-887e07fc9052
                        </ns3:HostedDataID>
                  </ns3:DataStorageItem>
            </ns3:StoreHostedData>
      </ns3:Action>
</ns4:IPGApiActionRequest>
```

A resposta contém as informações armazenadas. Por motivos de segurança, somente os 6 primeiros e os últimos 4 dígitos dos números de cartão de crédito são enviados de volta.

```
<ns4:IPGApiActionResponse
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns4:successfully>true</ns4:successfully>
      <ns4:DataStorageItem>
            <ns2:CreditCardData>
                  <ns3:CardNumber>403587...4977</ns3:CardNumber>
                  <ns3:ExpMonth>12</ns3:ExpMonth>
                  <ns3:ExpYear>12</ns3:ExpYear>
            </ns2:CreditCardData>
            <ns2:HostedDataID>
                  d56feaaf-2d96-4159-8fd6-887e07fc9052
```

```
            </ns2:HostedDataID>
      </ns4:DataStorageItem>
</ns4:IPGApiActionResponse>
```

Se o ID dos Dados Armazenados (Hosted Data ID) não existe, a resposta da API indica um erro:

```
<ns4:IPGApiActionResponse
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns4:successfully>true</ns4:successfully>
      <ns2:Error Code="SGSDAS-020301">
            <ns2:ErrorMessage>
                  Hosted data id:
                  6c814261-a843-49fb-bacd-1411d3780286 not found.
            </ns2:ErrorMessage>
      </ns2:Error>
</ns4:IPGApiActionResponse>
```

O valor *successfully* é *false* somente se o Data Vault não pode ser determinado porque a solicitação resultou em um erro.

8.6 Excluir registros existentes

A ação “Excluir” (Delete) permite remover registros de dados que não são mais necessários:

```
<ns4:IPGApiActionRequest
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1">
      <ns3:Action>
            <ns3:StoreHostedData>
                  <ns3:DataStorageItem>
                        <ns3:Function>delete</ns3:Function>
                        <ns3:HostedDataID>
                              9605c2d1-428c-4de2-940e-4bec4737ab5d
                        </ns3:HostedDataID>
                  </ns3:DataStorageItem>
            </ns3:StoreHostedData>
      </ns3:Action>
</ns4:IPGApiActionRequest>
```

Uma exclusão bem-sucedida será confirmada com a seguinte resposta:

```
<ns4:IPGApiActionResponse
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns4:successfully>true</ns4:successfully>
</ns4:IPGApiActionResponse>
```

9. 3D Secure Authentication

O 3D Secure é um mecanismo de autenticação projetado para reduzir fraudes e rejeições em relação às transações do cartão-não presente.

Com a nossa solução Connect (consulte o Guia de integração do Connect), podemos gerenciar os fluxos necessários para o processo de autenticação para você. Se você, no entanto, preferir lidar com este processo e os redirecionamentos necessários, a API do Web Service permite que você faça chamadas de API únicas para as etapas necessárias:

1. Você faz uma chamada de API para verificar se o titular do cartão está matriculado para participar de um programa 3D Secure
2. Para os casos em que o titular do cartão está inscrito, você redireciona seu cliente para o Servidor de Controle de Acesso do emissor do cartão (ACS) usando a URL que você recebeu na resposta para o seu pedido de verificação
3. Você recebe a resposta de autenticação do pagador do emissor do cartão, que inclui a confirmação codificada do status de autenticação e envia essas informações em uma segunda chamada de API para que possamos verificar a assinatura, decodificá-la e fornecer o resultado da autenticação
4. Você finalmente enviou a transação financeira (venda ou pré-autorização), fazendo referência à autenticação obtida com um ID de transação

Chamada API para o Passo 1

Para verificar se o cartão foi inscrito em um programa 3D Secure, você precisa enviar uma solicitação de verificação com um parâmetro AuthenticateTransaction definido como "true" e TxType = payerAuth.

O seguinte representa um exemplo de uma Solicitação de Verificação (VEReq):

```
<?xml version="1.0" encoding="UTF-8"?><SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns4:IPGApiOrderRequest
xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1">
            <ns2:Transaction>
                <ns2:CreditCardTxType>
                    <ns2:StoreId>120995000</ns2:StoreId>
                    <ns2:Type>payerAuth</ns2:Type>
                </ns2:CreditCardTxType>
                <ns2:CreditCardData>
                    <ns2:CardNumber>542606XXXXXX4979</ns2:CardNumber>
                    <ns2:ExpMonth>12</ns2:ExpMonth>
                    <ns2:ExpYear>24</ns2:ExpYear>
                    <ns2:CardCodeValue>XXX</ns2:CardCodeValue>
                </ns2:CreditCardData>
                <ns2:CreditCard3DSecure>
      <ns2:AuthenticateTransaction>true</ns2:AuthenticateTransaction>
                </ns2:CreditCard3DSecure>
                <ns2:Payment>
                    <ns2:ChargeTotal>13.99</ns2:ChargeTotal>
                    <ns2:Currency>986</ns2:Currency>
```

```
                </ns2:Payment>
            </ns2:Transaction>
        </ns4:IPGApiOrderRequest>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Nosso plug-in integrado (MPI) verifica a participação do cartão no diretório 3D Secure e retorna a URL de redirecionamento do Servidor de controle de acesso (ACS) do emissor do cartão.

Se o cartão estiver inscrito no 3D Secure, a resposta ao pedido de verificação deve conter os seguintes valores-chave:

- PaReq: o pedido de autenticação do portador, necessário para iniciar a URL da autenticação ACS: o destino do redirecionamento 3D Secure
- Term URL: a URL, que a ACS deve enviar o resultado no seu aplicativo
- MD: dados do estabelecimento que devem ser enviados para a URL da ACS.

O seguinte exemplo representa uma resposta VEReq:

```
<?xml version="1.0" encoding="UTF-8"?><SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ipgapi:IPGApiOrderResponse
xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi"
xmlns:a1="http://ipg-online.com/ipgapi/schemas/a1"
xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1">
            <ipgapi:ApprovalCode>?:waiting 3dsecure</ipgapi:ApprovalCode>
            <ipgapi:Brand>MASTERCARD</ipgapi:Brand>            <ipgapi:Comme
rcialServiceProvider>TELECASH</ipgapi:CommercialServiceProvider>
            <ipgapi:OrderId>A-4b9804e6410b84475809e59e1b26</ipgapi:OrderId>
            <ipgapi:IpgTransactionId>8383394822</ipgapi:IpgTransactionId>
            <ipgapi:PaymentType>CREDITCARD</ipgapi:PaymentType>
            <ipgapi:TDate>1493130774</ipgapi:TDate>
            <ipgapi:TDateFormatted>2017.04.25
16:32:54(CEST)</ipgapi:TDateFormatted>
            <ipgapi:TransactionTime>1493130774</ipgapi:TransactionTime>
            <ipgapi:Secure3DResponse>
                <v1:Secure3DVerificationResponse>
                    <v1:VerificationRedirectResponse>
                        <v1:AcsURL>https://3ds-
acs.test.modirum.com/mdpayacs/pareq</v1:AcsURL>
<v1:PaReq> c7fb83b8ag...73t4a827t4af8738a</v1:PaReq>
<v1:TermUrl>https://www.mywebshop.com/process3dSecure/</v1:TermUrl>
                        <v1:MD>MD1234....sdfk</v1:MD>
                    </v1:VerificationRedirectResponse>
                </v1:Secure3DVerificationResponse>
            </ipgapi:Secure3DResponse>
        </ipgapi:IPGApiOrderResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Chamada API para a Etapa 3

Depois de ter redirecionado o titular do cartão para autenticação e ter recebido a resposta de autenticação do emissor do cartão, você deve enviar os PARes e o MD em sua segunda chamada para a nossa API:

```
<?xml version="1.0" encoding="UTF-8"?><SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns4:IPGApiOrderRequest
xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1">
            <ns2:Transaction>
                <ns2:CreditCardTxType>
                    <ns2:StoreId>120995000</ns2:StoreId>
                    <ns2:Type>payerAuth</ns2:Type>
                </ns2:CreditCardTxType>
                <ns2:CreditCard3DSecure>
                    <ns2:Secure3DRequest>
                        <ns2:Secure3DAuthenticationRequest>
                            <ns2:AcsResponse>
                             <ns2:MD>MDasdadA5809e59e1b263b4aa9</ns2:MD>
                                                    <ns2:PaRes>eJzVWNey
q8iS…83IBmfhg</ns2:PaRes>
                            </ns2:AcsResponse>
                        </ns2:Secure3DAuthenticationRequest>
                    </ns2:Secure3DRequest>
                </ns2:CreditCard3DSecure>
                <ns2:Payment/>
                <ns2:TransactionDetails>
                    <ns2:IpgTransactionId>8383394822</ns2:IpgTransactionId>
                </ns2:TransactionDetails>
            </ns2:Transaction>
        </ns4:IPGApiOrderRequest>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Nosso Gateway verifica a resposta e fornece o resultado de volta para você, incluindo os dados necessários como parte da solicitação de autorização.

O seguinte exemplo representa uma resposta de autenticação:

```
<?xml version="1.0" encoding="UTF-8"?><SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ipgapi:IPGApiOrderResponse
xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi"
xmlns:a1="http://ipg-online.com/ipgapi/schemas/a1"
xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1">
            <ipgapi:ApprovalCode>Y:ECI2/5:Authenticated</ipgapi:ApprovalCode
>
            <ipgapi:Brand>MASTERCARD</ipgapi:Brand>
            <ipgapi:CommercialServiceProvider>TELECASH</ipgapi:CommercialSer
viceProvider>
            <ipgapi:OrderId>A-123456789</ipgapi:OrderId>
            <ipgapi:IpgTransactionId>8383394827</ipgapi:IpgTransactionId>
            <ipgapi:PaymentType>CREDITCARD</ipgapi:PaymentType>
```

```
            <ipgapi:TDate>1493137253</ipgapi:TDate>
            <ipgapi:TDateFormatted>2017.04.25 18:20:53
(CEST)</ipgapi:TDateFormatted>
            <ipgapi:TransactionResult>APPROVED</ipgapi:TransactionResult>
            <ipgapi:TransactionTime>1493137253</ipgapi:TransactionTime>
            <ipgapi:Secure3DResponse>
                <v1:ResponseCode3dSecure>1</v1:ResponseCode3dSecure>
            </ipgapi:Secure3DResponse>
        </ipgapi:IPGApiOrderResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Chamada API para o Passo 4

O seguinte exemplo representa uma transação Sale efetuada após solicitação previamente autenticada:

```
<soap:Envelope xmlns:Soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:IPGApiOrderRequest
xmlns="http://ipg-online.com/ipgapi/schemas/a1"
xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
xmlns:ns3="http://ipg-online.com/ipgapi/schemas/ipgapi"
      <ns2:Transaction>
        <ns2:CreditCardTxType>
          <ns2:StoreId>110995000</ns2:StoreId>
          <ns2:Type>sale</ns2:Type>
        </ns2:CreditCardTxType>
        <ns2:TransactionDetails>
                <ns2:IpgTransactionId>8383383800</ns2:IpgTransactionId>
        </ns2:TransactionDetails>
      </ns2:Transaction>
    </ns3:IPGApiOrderRequest>
  </soap:Body>
</soap:Envelope>
```

Se você tiver o seu próprio Plug-in (MPI) para o 3D Secure ou usar um provedor de terceiros para isso, você pode alternativamente enviar o resultado do processo de autenticação na sua mensagem de transação de venda ou pré-autorização para a API do Web Service. Veja os elementos *CreditCard3DSecure* no capítulo 9 Visão geral de tags XML deste documento.

Em princípio, pode ocorrer que as autenticações 3D Secure não possam ser processadas com sucesso por razões técnicas. Se um dos sistemas envolvidos no processo de autenticação não estiver respondendo temporariamente, a transação de pagamento será processada como uma transação de comércio eletrônico "regular" (ECI 7). **Uma mudança de responsabilidade para o emissor do cartão para possíveis rejeições não está garantida neste caso**. Se você preferir que essas transações não sejam processadas, nossa equipe de suporte técnico pode bloqueá-las para sua loja, mediante solicitação.

10. Visão geral de tags de XML

Visão geral por tipo de transação

A seguir, mostraremos quais tags de XML precisam ser enviadas para cada tipo de transação, bem como as que podem ser usadas opcionalmente. Use apenas os campos definidos abaixo e observe a ordem.

Para tags XML relacionadas às transações de cartão presente em uma solução de captura com um leitor de chip e senha, consulte os xsd's no apêndice deste documento.

Abreviações:

m:	obrigatório
o:	opcional
d:	opcional com valor padrão
a e **b**:	máximo um de dois valores
1:	Se **a** ou **b** for informado,opcional obrigatório se **a** e **b** não foram informados
3:	obrigatório para transações 3D Secure
s:	consulte detalhes no capítulo de 3D Secure
f:	obrigatório para transações VISA de instituições financeiras baseadas no Reino Unido com o MCC 6012
r:	obrigatório para transações recorrentes SEPA Direct Debit
p:	obrigatório para split shipment

Caminho/ Nome	**Cartão de crédito**							**Débito**			
todos os caminhos relacionados a `ipgapi:IPGApiOrderRequest/ v1:Transaction`	Venda	ForceTicket	PreAuth	Postauth	Cancelame	Credit	Estorno	Venda	Cancelame	Credit	Estorno
v1:CreditCardTxType/ v1:Type	**m**	**m**	**m**	**m**	**m**	**m**	**m**				
v1:CreditCardData/ v1:CardNumber	**a**	**a**	**a**			**a**					
v1:CreditCardData/ v1:ExpMonth	**a**	**a**	**a**			**a**					
v1:CreditCardData/ v1:ExpYear	**a**	**a**	**a**			**a**					
v1:CreditCardData/ v1:CardCodeValue	*o*	*o*	*o*			*o*					
v1:CreditCardData/ v1:TrackData	**b**	**b**	**b**			**b**					
v1:CreditCardData/ v1:Brand	*o*	*o*	*o*			*o*					
v1:CreditCard3DSecure/ v1:VerificationResponse	3	3	3			3					
v1:CreditCard3DSecure/ v1:PayerAuthenticationResponse	s	s	s			s					
v1:CreditCard3DSecure/ v1:DRSPECI	s	s	s			s					

v1:CreditCard3DSecure/ v1:AuthenticationValue	s	s	s			s					
v1:CreditCard3DSecure/ v1:XID	s	s	s			s					
v1:CreditCard3DSecure/ v1:AuthenticateTransaction	s	s	s			s					
v1:CreditCard3DSecure/ v1:Secure3DRequest/ v1: Secure3DAuthenticationRequest/ v1: IVRAuthenticationRequest	s	s	s			s					
v1:CreditCard3DSecure/ v1:Secure3DRequest/ v1:Secure3DAuthenticationRequest/ v1:AcsResponse	s	s	s			s					
v1:CreditCard3DSecure/ v1:Secure3DRequest/ v1: Secure3DVerificationRequest v1: IVRVerificationRequest	s	s	s			s					
v1:CreditCard3DSecure/ v1:Secure3DverificationResponse/ v1:IVRVerificationResponse	s	s	s			s					
v1:CreditCard3DSecure/ v1:Secure3DverificationResponse/ v1:VerificationRedirectResponse	s	s	s			s					
v1:cardFunction/ v1:Type	*o*	*o*	*o*			*o*					
v1:DE_DirectDebitTxType/ v1:Type								**m**	**m**	**m**	**m**
v1:DE_DirectDebitData/ v1:BIC								*o*		**1**	
v1:DE_DirectDebitData/ v1:IBAN								**a**		**a**	
v1:DE_DirectDebitData/ v1:TrackData								**b**		**b**	
v1:DE_DirectDebitData/ v1:MandateReference								**m**			
v1:DE_DirectDebitData/ v1:MandateType								*d,r*			
v1:DE_DirectDebitData/ v1:DateOfMandate								*r*			
v1:Payment/ v1:HostedDataID	***1***	***1***	***1***			***1***		***1***		***1***	
v1:Payment/ v1:HostedDataStoreID	***1***	***1***	***1***			***1***		***1***		***1***	
v1:Payment/ v1:DeclineHostedDataDuplicates	***1***	***1***	***1***			***1***		***1***		***1***	
v1:Payment/ v1:numberOfInstallments	*o*										
v1:Payment/ v1:installmentsInterest	*d*										
v1:Payment/ v1:installmentDelayMonths	*o*										
v1:Payment/ v1:SubTotal	*o*	*o*	*o*	*o*	*o*	*o*		*o*	*o*	*o*	
v1:Payment/ v1:ValueAddedTax	*o*	*o*	*o*	*o*	*o*	*o*		*o*	*o*	*o*	
v1:Payment/ v1:localTax	*o*	*o*	*o*	*o*	*o*	*o*		*o*	*o*	*o*	

v1:Payment/ v1:DeliveryAmount	*o*	*o*	*o*	*o*	*o*	*o*		*o*	*o*	*o*	
v1:Payment/ v1:ChargeTotal	**m**	**m**	**m**	**m**	**m**	**m**		**m**	**m**	**m**	
v1:Payment/ v1:Currency	**m**	**m**	**m**	**m**	**m**	**m**		**m**	**m**	**m**	
v1:recurringType	*o*										
v1:WalletType	*o*										
v1:WalletID	*o*										
v1:TransactionDetails/ v1:OrderId	*o*	*o*	*o*	**m**	**m**	*o*	**a**	*o*	**m**	*o*	**a**
v1:TransactionDetails/ v1:MerchantTransactionId	*o*	*o*	*o*	*o*	*o*	*o*	*o*	*o*	*o*	*o*	*o*
v1:TransactionDetails/ v1:Ip	*o*		*o*			*o*		*o*		*o*	
v1:TransactionDetails/ v1:ReferenceNumber		**m**									
v1:TransactionDetails/ v1:Tdate							**a**				**a**
v1:TransactionDetails/ v1:ReferencedMerchantTransactionId							**b**				**b**
v1:TransactionDetails/ v1:TransactionOrigin	*d*		*d*			*d*					
v1:TransactionDetails/ v1:InvoiceNumber	*o*	*o*	*o*	*o*		*o*		*o*		*o*	
v1:TransactionDetails/ v1:PONumber	*o*	*o*	*o*			*o*		*o*		*o*	
v1:TransactionDetails/ v1:DynamicMerchantName	*o*	*o*	*o*			*o*		*o*		*o*	
v1:TransactionDetails/ v1:Comments	*o*	*o*	*o*	*o*	*o*	*o*	*o*	*o*	*o*	*o*	*o*
v1:TransactionDetails/ v1:Terminal/ v1:TerminalID	*o*	*o*	*o*			*o*		*o*		*o*	
v1:TransactionDetails/ v1:InquiryRateReference	*o*	*o*	*o*								
v1:TransactionDetails/ v1:SplitShipment/ v1:SequenceCount			*o*	*o*							
v1:TransactionDetails/ v1:SplitShipment/ v1:FinalShipment				**p**							
v1:Billing/ v1:CustomerID	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Billing/ v1:Name	*o*	*o*	*o*			*o*		**m**		**m**	
v1:Billing/ v1:Company	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Billing/ v1:Address1	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Billing/ v1:Address2	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Billing/ v1:City	*o*	*o*	*o*			*o*		*o*		*o*	

v1:Billing/ v1:State	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Billing/ v1:Zip	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Billing/ v1:Country	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Billing/ v1:Phone	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Billing/ v1:Fax	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Billing/ v1:Email	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Shipping/ v1:Type	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Shipping/ v1:Name	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Shipping/ v1:Address1	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Shipping/ v1:Address2	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Shipping/ v1:City	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Shipping/ v1:State	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Shipping/ v1:Zip	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Shipping/ v1:Country	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Basket/ v1:Item/ v1:ID	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Basket/ v1:Item/ v1:Description	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Basket/ v1:Item/ v1:SubTotal											
v1:Basket/ v1:Item/ v1:ValueAddedTax											
v1:Basket/ v1:Item/ v1:DeliveryAmount											
v1:Basket/ v1:Item/ v1:ChargeTotal	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Basket/ v1:Item/ v1:Currency											
v1:Basket/ v1:Item/ v1:Quantity	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Basket/ v1:Item/ v1:Option/ v1:Name	*o*	*o*	*o*			*o*		*o*		*o*	
v1:Basket/ v1:Item/ v1:Choice	*o*	*o*	*o*			*o*		*o*		*o*	

v1:TopUpTxType/ v1:MPCharge/ v1:MNSP											
v1:TopUpTxType/ v1:MPCharge/ v1:MSISDN											
v1:TopUpTxType/ v1:MPCharge/ v1:PaymentType											
v1:ClientLocale/ v1:Language	*d*	*d*	*d*	*d*	*d*	*d*	*d*	*d*	*d*	*d*	*d*
v1:ClientLocale/ v1:Country	*d*	*d*	*d*	*d*	*d*	*d*	*d*	*d*	*d*	*d*	*d*
v1:MCC6012Details/ v1:BirthDate	**f**	**f**	**f**								
v1:MCC6012Details/ v1:AccountFirst6	**f,a**	**f,a**	**f,a**								
v1:MCC6012Details/ v1:AccountLast4	**f,a**	**f,a**	**f,a**								
v1:MCC6012Details/ v1:AccountNumber	**f,b**	**f,b**	**f,b**								
v1:MCC6012Details/ v1:PostCode	**f**	**f**	**f**								
v1:MCC6012Details/ v1:Surname	**f**	**f**	**f**								

Caminho/ Nome	**PayPal**				**Mobile Top-up**
todos os caminhos relacionados a `ipgapi:IPGApiOrderRequest/ v1:Transaction`	Postauth	Cancelame	Credit	Estorno	MPCharge
v1:CreditCardTxType/ v1:Type					
v1:CreditCardData/ v1:CardNumber					
v1:CreditCardData/ v1:ExpMonth					
v1:CreditCardData/ v1:ExpYear					
v1:CreditCardData/ v1:CardCodeValue					
v1:CreditCardData/ v1:TrackData					
v1:CreditCard3DSecure/ v1:VerificationResponse					
v1:CreditCard3DSecure/ v1:PayerAuthenticationResponse					
v1:CreditCard3DSecure/ v1:AuthenticationValue					
v1:CreditCard3DSecure/ v1:XID					
v1:DE_DirectDebitTxType/ v1:Type					

v1:DE_DirectDebitData/ v1:BIC					
v1:DE_DirectDebitData/ v1:IBAN					
v1:DE_DirectDebitData/ v1:MandateReference					
v1:DE_DirectDebitData/ v1:MandateType					
v1:DE_DirectDebitData/ v1:TrackData					
v1:PayPalTxType/ v1:Type	**m**	**m**	**m**	**m**	
v1:Payment/ v1:HostedDataID					
v1:Payment/ v1:HostedDataStoreID					
v1:Payment/ v1:DeclineHostedDataDuplicates					
v1:Payment/ v1:SubTotal	*o*	*o*	*o*		*o*
v1:Payment/ v1:ValueAddedTax	*o*	*o*	*o*		*o*
v1:Payment/ v1:DeliveryAmount	*o*	*o*	*o*		*o*
v1:Payment/ v1:ChargeTotal	**m**	**m**	**m**		**m**
v1:Payment/ v1:Currency	**m**	**m**	**m**		**m**
v1:TransactionDetails/ v1:OrderId	**m**	**m**	*o*	**m**	*o*
v1:TransactionDetails/ v1:MerchantTransactionId	*o*	*o*	*o*	*o*	*o*
v1:TransactionDetails/ v1:Ip			*o*		*o*
v1:TransactionDetails/ v1:ReferenceNumber					
v1:TransactionDetails/ v1:Tdate				**a**	
v1:TransactionDetails/ v1:ReferencedMerchantTransactionId				**b**	
v1:TransactionDetails/ v1:TransactionOrigin			*d*		
v1:TransactionDetails/ v1:InvoiceNumber			*o*		*o*
v1:TransactionDetails/ v1:PONumber			*o*		*o*
v1:TransactionDetails/ v1:DynamicMerchantName			*o*		*o*
v1:TransactionDetails/ v1:Comments	*o*	*o*	*o*	*o*	*o*
v1:Billing/ v1:CustomerID			*o*		*o*
v1:Billing/ v1:Name			*o*		*o*
v1:Billing/ v1:Company			*o*		*o*

v1:Billing/ v1:Address1			*o*		*o*
v1:Billing/ v1:Address2			*o*		*o*
v1:Billing/ v1:City			*o*		*o*
v1:Billing/ v1:State			*o*		*o*
v1:Billing/ v1:Zip			*o*		*o*
v1:Billing/ v1:Country			*o*		*o*
v1:Billing/ v1:Phone			*o*		*o*
v1:Billing/ v1:Fax			*o*		*o*
v1:Billing/ v1:Email			**m**		*o*
v1:Shipping/ v1:Type			*o*		
v1:Shipping/ v1:Name			*o*		
v1:Shipping/ v1:Address1			*o*		
v1:Shipping/ v1:Address2			*o*		
v1:Shipping/ v1:City			*o*		
v1:Shipping/ v1:State			*o*		
v1:Shipping/ v1:Zip			*o*		
v1:Shipping/ v1:Country			*o*		
v1:Basket/ v1:Item/ v1:ID			*o*		
v1:Basket/ v1:Item/ v1:Description			*o*		
v1:Basket/ v1:Item/ v1:SubTotal					
v1:Basket/ v1:Item/ v1:ValueAddedTax					
v1:Basket/ v1:Item/ v1:DeliveryAmount					
v1:Basket/ v1:Item/ v1:ChargeTotal			*o*		
v1:Basket/ v1:Item/ v1:Currency					
v1:Basket/ v1:Item/ v1:Quantity			*o*		

v1:Basket/ v1:Item/ v1:Option/ v1:Name			*o*		
v1:Basket/ v1:Item/ v1:Choice			*o*		
v1:TopUpTxType/ v1:MPCharge/ v1:MNSP					**m**
v1:TopUpTxType/ v1:MPCharge/ v1:MSISDN					**m**
v1:TopUpTxType/ v1:MPCharge/ v1:PaymentType					**m**
v1:ClientLocale/ v1:Language	*d*	*d*	*d*	*d*	*d*
v1:ClientLocale/ v1:Country	*d*	*d*	*d*	*d*	*d*

10.2 Descrição das tags de XML

10.2.1 CreditCardTxType

Caminho/Nome	Tipo de esquema XML	Descrição
v1:CreditCardTxType/ v1:Type	xs:string	Armazena o tipo de transação. Os valores possíveis são `sale`, `forceTicket`, `preAuth`, `postAuth`, `return`, `credit` e `void`.

10.2.2 CreditCardData

Caminho/Nome	Tipo de esquema XML	Descrição
v1:CreditCardData/ v1:CardNumber	xs:string	Armazena o número de cartão de crédito do cliente. A string deve conter somente dígitos, ou seja, passar o número no formato *xxxx-xxxx-xxxx-xxxx*, por exemplo, resultará em um erro informado pela Web Service API.
v1:CreditCardData/ v1:ExpMonth	xs:string	Armazena o mês de vencimento do cartão de crédito do cliente. O conteúdo deste elemento sempre contém dois dígitos, ou seja, um cartão que vence em julho terá o elemento com o valor `07`.
v1:CreditCardData/ v1:ExpYear	xs:string	Armazena o ano de vencimento do cartão de crédito do cliente. As mesmas restrições de formato do elemento `v1:ExpMonth` aplicam-se aqui.
v1:CreditCardData/ v1:CardCodeValue	xs:string	Armazena o código de segurança de três ou quatro dígitos, às vezes chamado de código ou valor de verificação (CVV, CVC ou CSC), que normalmente está impresso no verso do cartão de crédito.
v1:CreditCardData/ v1:TrackData	xs:string	Armazena os dados da tarja de um cartão ao usar um leitor de cartão em vez de inserir os dados do cartão (opcionalmente, pode ser usado em vez de transmitir CardNumber, ExpMonth e ExpYear). Este campo precisa conter pelo menos os dados 1 e 2 de concatenação. Os dados de trilha 3 são opcionais. Os dados da trilha devem incluir os separadores de trilha e campo como eles são armazenados no cartão. Exemplo para o separador de dados de trilha dos dados de trilha 1 e 2 sem os dados: % ...?; ...?
v1:CreditCardData/ v1:TrackData	xs:string	Campo opcional para a bandeira do cartão de crédito. Se esse campo estiver habilitado, a transação somente será processada se o número do cartão corresponder à bandeira.

Para tags XML relacionadas às transações de cartão presente em uma solução de captura com um leitor de chip e senha, consulte os xsd's no apêndice deste documento.

10.2.3 recurringType

Caminho/Nome	Tipo de esquema XML	Descrição
v1:recurringType	xs:string	Este campo permite que você sinalize as transações como recorrentes. Pode ser configurado para FIRST para a primeira transação de uma série e REPEAT para as transações subseqüentes de uma série.

10.2.4 Wallet

Caminho/Nome	Tipo de esquema XML	Descrição
v1:Wallet/ v1:WalletType	xs:string	Este campo permite que você envie o tipo de carteira para as transações que foram iniciadas através de uma carteira digital. Atualmente, o único valor válido é MASTERPASS.
v1:Wallet/ v1:WalletID	xs:string	Este campo permite que você envie o ID da carteira para as transações que foram iniciadas através de uma carteira digital.

10.2.5 cardFunction

Path/Nome	Tipo de esquema XML	Descrição
`v1:cardFunction/ v1:Type`	`xs:string`	Este campo permite indicar a função do cartão no caso de cartões múltiplos (Combo), com crédito e débito no mesmo cartão. Podem ser usados `credit` ou `debit`.

10.2.6 CreditCard3DSecure

Caminho/Nome	Tipo de esquema XML	Descrição
`v1:CreditCard3DSecure/ v1:VerificationResponse`	`xs:string`	Armazena a VerificationResponse (VERes) do seu plug-in de Estabelecimento.
`v1:CreditCard3DSecure/ v1:PayerAuthenticationResponse`	`xs:string`	Armazena a PayerAuthenticationResponse (PARes) do seu plug-in de Estabelecimento.
`v1:CreditCard3DSecure/ v1:DSRPECI`	`xs:string`	Para definir o valor ECI para Digital Secure Remote Payments. Se você enviar este parâmetro, quaisquer valores para parâmetros VerificationResponse e PayerAuthenticationResponse serão ignorados.

v1:CreditCard3DSecure/ v1:AuthenticationValue	xs:string	Armazena o AuthenticationValue (MasterCard: AAV ou VISA: CAAV) de seu plug-in de Estabelecimento.
v1:CreditCard3DSecure/ v1:XID	xs:string	Armazena o XID do seu plug-in de Estabelecimento.
v1:CreditCard3DSecure/ v1:AuthenticateTransaction	xs:boolean	Indica se a transação será autenticada como transação 3DSecure. Se um cartão estiver inscrito, a resposta conterá o elemento de resposta de redirecionamento de verificação.

Observe que esses valores são recebidos pelo seu próprio Plug-in de Estabelecimento para o 3D Secure ou uma solução de um provedor de 3D Secure. A funcionalidade integrada 3D Secure do recurso Connect não pode ser usada para transações via API por motivos técnicos.

10.2.7 Autenticação 3D Secure/ Resposta de Redirecionamento de Verificação

Caminho/Nome	Tipo de esquema XML	Descrição
v1:VerificationRedirectResponse v1:AcsURL	xs:string	Representa o endereço do redirecionamento 3D Secure.
v1:VerificationRedirectResponse v1:PaReq	xs:string	Representa os dados PAReq que devem ser enviados no atributo "PAReq" para a URL da ACS.
v1:IVRVerificationRequest/ v1:TermUrl	xs:string	Representa o padrão TermURL, que deve ser usado para processar a resposta do processo 3D Secure. No caso de um estabelecimento desejar analisar a resposta por si mesmo, ele deve especificar o parâmetro "TermUrl" no formulário com seu URL personalizado, no qual ele irá processar a resposta e chamar a API com a resposta PARes e Merchant Data.
v1:IVRVerificationRequest/ v1:MD	xs:string	Representa os dados do estabelecimento que devem ser enviados no atributo "MD" para a URL da ACS.

10.2.8 Autenticação 3D Secure/ Resposta ACS

Caminho/Nome	Tipo de esquema XML	Descrição
v1:AcsResponse v1:MD	xs:string	Merchant Data do atributo POST do redirecionamento da ACS (atributo "MD").
v1:AcsResponse v1:PaRes	xs:string	Representa os dados PARes do atributo POST do redirecionamento da ACS (atributo "PARes").

10.2.9 Pagamento

Caminho/Nome	Tipo de esquema XML	Descrição
v1:Payment/ v1:HostedDataID	xs:string	Armazena o ID dos dados armazenados para o produto do Data Vault
v1:Payment/ v1:HostedDataStoreID	xs:string	Armazena o ID dos dados armazenados para o produto do Data Vault nesta loja (somente como usuário técnico)
v1:Payment/ v1:DeclineHostedDataDuplicates	xs:string	Recusa contas duplicadas de cartão de crédito
v1:Payment/ v1:numberOfInstallments	xs:string	Armazena o número de parcelas para uma transação de Venda (Sale) parcelada.
v1:Payment/ v1:installmentsInterest	xs:string	Indica se aplica juros na transação de Venda (Sale) parcelada; Valores possíveis "sim" ou "não".
v1:Payment/ v1:installmentDelayMonths	xs:string	Representa o número de meses em que o primeiro pagamento será adiado; Valores possíveis no intervalo <1; 99>
v1:Payment/ v1:SubTotal	xs:decimal	Armazena o subtotal de um pedido. Se este membro estiver configurado, então o ChargeTotal também deverá estar.
v1:Payment/ v1:ValueAddedTax	xs:decimal	Armazena o IVA de um pedido. Se este membro estiver configurado, então o Subtotal também deverá estar .*não aplicável ao Brasil
v1:Payment/ v1:DeliveryAmount	xs:decimal	Armazena o valor de entrega de um pedido. Se este membro estiver configurado, então o Subtotal também deverá estar.
v1:Payment/ v1:ChargeTotal	xs:double	Armazena o valor da transação. O número de posições após o ponto decimal não deve exceder 2, por exemplo, 3.123 seria um valor inválido, porém, 3.12, 3.1 e 3 estão corretos.
v1:Payment/ v1:Currency	xs:string	Armazena a moeda como um valor ISO 4217 de três dígitos. Por exemplo, 986 para BRL – Reais (R$)

10.2.10 TransactionDetails

Caminho/Nome	Tipo de esquema XML	Descrição
v1:TransactionDetails/ v1:OrderId	xs:string	Armazena o Número do Pedido (Order ID). Ele deve ser exclusivo por ID da Loja. Se o Número do Pedido (Order ID) não for transmitido, o Sicredi e-commerce gerará um automaticamente.
v1:TransactionDetails/ v1:MerchantTransactionId	xs:string	Permite atribuir um ID exclusivo para a transação. Essa identificação pode ser usada para fazer referência a essas transações em uma solicitação Void (ReferencedMerchantTransactionId) ou para recuperar os detalhes da transação com a ação da API InquiryTransaction. O ID exclusivo precisa ser aplicado pelo estabelecimento.
v1:TransactionDetails/ v1:Ip	xs:string	Armazena o endereço IP do cliente que pode ser usado pelo Web Service API para detecção de fraude por endereço IP. O IP deve ser informado no formato *xxx.xxx.xxx.xxx*, ou seja, `128.0.10.2` seria um IP válido.
v1:TransactionDetails/ v1:ReferenceNumber	xs:string	Armazena o número de referência de seis dígitos que você recebeu resultante da uma autorização externa bem-sucedida (por ex., por telefone). O Sicredi e-commerce precisa desse número para mapear exclusivamente uma transação de *ForceTicket* para uma autorização externa realizada anteriormente.
v1:TransactionDetails/ v1:TDate	xs:string	Armazena a TDate da transação de *Venda à vista (Sale)*, *PostAuth (Captura Posterior)*, *ForceTicket* ou *Cancelamento (return)* ou a qual essa transação de *Estorno (Void)* se refere. Um valor de TDate é retornado dentro da resposta para uma transação bem-sucedida de um desses quatro tipos. Ao realizar uma transação de *Estorno*, você precisa passar a TDate além do Numero do Pedido (Order ID) para identificação exclusiva para a transação a ser estornada. O cenário apresentado abaixo serve de exemplo.
v1:TransactionDetails/ v1:ReferencedMerchantTransactionId	xs:string	Armazena o MerchantTransactionId da Venda à vista (Sale), PostAuth (Captura Posterior), ForceTicket, Cancelamento (return) ou a qual essa transação Estorno (Void) se refere. Isso pode ser usado como uma alternativa ao TDate se você atribuir um MerchantTransactionId na solicitação de transação original.

v1:TransactionDetails/ v1:TransactionOrigin	xs:string	A origem da transação. Os valores possíveis são ECI (se o pedido foi recebido por Internet), MOTO (pedido por correios ou telefone) e RETAIL (pessoalmente).
v1:TransactionDetails/ v1:SplitShipment/ v1:SequenceCount	xs:int	Armazena o número total de remessas em caso de envio dividido. Pode ser incluído no PreAuth ou no primeiro PostAuth. Um valor diferente no primeiro PostAuth substitui o valor do PreAuth.
v1:TransactionDetails/ v1:SplitShipment/ v1:FinalShipment	xs:boolean	Precisa ser definido como "true" no PostAuth final de uma série de envios divididos.
v1:TransactionDetails/ v1:InvoiceNumber	xs:string	Armazena o número da fatura.
v1:TransactionDetails/ v1:PONumber	xs:string	Armazena o número de pedido da compra.
v1:TransactionDetails/ v1:DynamicMerchantName	xs:string	Armazena um nome dinâmico de estabelecimento para o extrato do portador de cartão (Soft Descriptor)
v1:TransactionDetails/ v1:Comments	xs:string	Armazena os comentários.

10.2.11 Cobrança

Caminho/Nome	Tipo de esquema XML	Descrição
v1:Billing/ v1:CustomerID	xs:string	Armazena o ID para seu cliente.
v1:Billing/ v1:Name	xs:string	Armazena o nome do cliente. Se informado, aparecerá nos seus relatórios de transação.
v1:Billing/ v1:Company	xs:string	Armazena a empresa do cliente. Se informado, aparecerá nos seus relatórios de transação.
v1:Billing/ v1:Address1	xs:string	Armazena a primeira linha do endereço do cliente. Se informado, aparecerá nos seus relatórios de transação.
v1:Billing/ v1:Address2	xs:string	Armazena a segunda linha do endereço do cliente. Se informado, aparecerá nos seus relatórios de transação.
v1:Billing/ v1:City	xs:string	Armazena a cidade do cliente. Se informado, aparecerá nos seus relatórios de transação.
v1:Billing/ v1:State	xs:string	Armazena o estado do cliente. Se informado, aparecerá nos seus relatórios de transação.
v1:Billing/ v1:Zip	xs:string	Armazena o CEP do cliente. Se informado, aparecerá nos seus relatórios de transação.
v1:Billing/ v1:Country	xs:string	Armazena o país do cliente. Se informado, aparecerá nos seus relatórios de transação.
v1:Billing/ v1:Phone	xs:string	Armazena o número de telefone do cliente. Se informado, aparecerá nos seus relatórios de transação.

v1:Billing/ v1:Fax	xs:string	Armazena o número de fax do cliente. Se informado, aparecerá nos seus relatórios de transação.
v1:Billing/ v1:Email	xs:string	Armazena o endereço de e-mail do cliente. Se informado, aparecerá nos seus relatórios de transação. Se você estiver usando o recurso de notificações de transação por e-mail, esse endereço de e- mail será usado para notificar seu cliente.

10.2.12 Envio (Entrega)

Caminho/Nome	Tipo de esquema XML	Descrição
v1:Shipping/ v1:Type	xs:string	Armazena a forma de entrega.
v1:Shipping/ v1:Name	xs:string	Armazena o nome do destinatário. Se informado, aparecerá nos seus relatórios de transação.
v1:Shipping/ v1:Address1	xs:string	Armazena a primeira linha do endereço de envio. Se informado, aparecerá nos seus relatórios de transação.
v1:Shipping/ v1:Address2	xs:string	Armazena a segunda linha do endereço de envio. Se informado, aparecerá nos seus relatórios de transação.
v1:Shipping/ v1:City	xs:string	Armazena a cidade do destinatário. Se informado, aparecerá nos seus relatórios de transação.
v1:Shipping/ v1:State	xs:string	Armazena o estado do destinatário. Se informado, aparecerá nos seus relatórios de transação.
v1:Shipping/ v1:Zip	xs:string	Armazena o CEP do destinatário. Se informado, aparecerá nos seus relatórios de transação.
v1:Shipping/ v1:Country	xs:string	Armazena o país do destinatário. Se informado, aparecerá nos seus relatórios de transação.

10.2.13 ClientLocale

Caminho/Nome	Tipo de esquema XML	Descrição
v1:ClientLocale/ v1:Language	xs:string	Se você estiver usando o recurso de notificações de transação por e-mail, este idioma será usado para notificar seu cliente. Os valores possíveis são: `de, en, it, pt.`
v1:ClientLocale/ v1:Country	xs:string	Especifica a variante do idioma. Este membro somente pode ser definido se o idioma for definido. Os valores possíveis são: `DE, GB, IT, BR.` Se você não definir um país, um país correspondente será escolhido.

Se você não enviar as informações de idioma na transação, as configurações de idioma da sua loja serão usadas para os e-mails de notificação.

11. Como criar uma mensagem de solicitação SOAP

Depois de criar sua transação em XML, é preciso criar uma mensagem de solicitação SOAP descrevendo a chamada da operação do Web Service que você deseja realizar. Isso significa que enquanto a transação codificada com XML que você estabeleceu (conforme descrito no capítulo anterior) representa o argumento da operação, a mensagem de solicitação SOAP codifica a chamada de operação real.
Criar uma mensagem de solicitação SOAP é uma tarefa muito objetiva. A mensagem de SOAP completa envolvendo as transações de *Venda* (Sale) de XML tem o seguinte formato:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
            xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
      <SOAP-ENV:Header />
      <SOAP-ENV:Body>
            <ipgapi:IPGApiOrderRequest
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
                  <v1:Transaction>
                        <v1:CreditCardTxType>
                              <v1:Type>sale</v1:Type>
                        </v1:CreditCardTxType>
                        <v1:CreditCardData>
                              <v1:CardNumber>
                                    4111111111111111
                              </v1:CardNumber>
                               <v1:ExpMonth>12</v1:ExpMonth>
                               <v1:ExpYear>07</v1:ExpYear>
                        </v1:CreditCardData>
                        <v1:Payment>
                               <v1:ChargeTotal>19.00</v1:ChargeTotal>
                               <v1:Currency>986</v1:Currency>
                        </v1:Payment>
                  </v1:Transaction>
            </ipgapi:IPGApiOrderRequest>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Em resumo, a mensagem de solicitação SOAP contém um envelope SOAP composto por um cabeçalho e um corpo. Apesar de entradas de cabeçalho específicas não serem exigidas para chamar o Web Service, o corpo SOAP usa o documento de XML de transação como um subelemento, conforme mostrado acima. Observe que não há mais requisitos para transações de outro tipo além de *Venda (Sale)*. Isso significa que o formato geral da mensagem de solicitação SOAP, não importando o tipo de transação atual, é o seguinte:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
            xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
      <SOAP-ENV:Header />
      <SOAP-ENV:Body>
            <ipgapi:IPGApiOrderRequest
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1">
                  <v1:Transaction>
                        <!-- transaction content -->
                  </v1:Transaction>
            </ipgapi:IPGApiOrderRequest>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Por fim, você talvez tenha percebido que não há entradas específicas que descrevam qual operação do Web Service deverá ser chamada. Na realidade, o Sicredi e-commerce automaticamente mapeia o elemento `ipgapi:IPGApiOrderRequest` para a operação do Web Service correspondente.

11.1 Como ler a mensagem de resposta SOAP

A mensagem de resposta SOAP pode ser entendida como resultado da operação do Web Service. Assim, o processamento da mensagem de solicitação SOAP pode ter resultado tanto em uma mensagem de resposta SOAP se houve êxito (ou seja, o parâmetro de retorno) ou em uma mensagem de falha de SOAP em caso de falha (*thrown exception*). Os dois tipos de mensagem de SOAP estão no corpo da mensagem de resposta de HTTP.

11.1.1 Mensagem de resposta SOAP

Uma mensagem de resposta SOAP é recebida como resultado da aprovação da transação. Ela sempre apresenta o seguinte esquema:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
            xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
      <SOAP-ENV:Header />
      <SOAP-ENV:Body>
            <ipgapi:IPGApiOrderResponse
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
                  <!-- transaction result -->
            </ipgapi:IPGApiOrderResponse>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Se você envia uma Ação, obtém `ipgapi:IPGApiActionResponse.`

Novamente, não há cabeçalhos definidos. O corpo de SOAP contém o resultado real da transação, contido no elemento `ipgapi:IPGApiOrderResponse` ou `ipgapi:IPGApiOrderRequest`. Seus subelementos e significados serão apresentados no próximo capítulo. Contudo, para dar um breve exemplo, uma transação de *Venda* (Sale) aprovada é envolvida em uma mensagem de SOAP similar ao exemplo a seguir:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
            xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
      <SOAP-ENV:Header />
      <SOAP-ENV:Body>
            <ipgapi:IPGApiOrderResponse
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
                  <ipgapi:CommercialServiceProvider>
                        BNLP
                  </ipgapi:CommercialServiceProvider>
                  <ipgapi:TransactionTime>
                        1192111687392
                  </ipgapi:TransactionTime>
                  <ipgapi:ProcessorReferenceNumber>
                        3105
                  </ipgapi:ProcessorReferenceNumber>
                  <ipgapi:ProcessorResponseMessage>
                        Function performed error-free
                  </ipgapi:ProcessorResponseMessage>
                  <ipgapi:ErrorMessage />
                  <ipgapi:OrderId>
                        62e3b5df-2911-4e89-8356-1e49302b1807
                  </ipgapi:OrderId>
                  <ipgapi:ApprovalCode>
                        Y:440368:0000057177:PPXM:0043364291
                  </ipgapi:ApprovalCode>
                  <ipgapi:AVSResponse>PPX</ipgapi:AVSResponse>
                  <ipgapi:TDate>1192140473</ipgapi:TDate>
                  <ipgapi:TransactionResult>
                        APPROVED
                  </ipgapi:TransactionResult>
                  <ipgapi:TerminalID>123456</ipgapi:TerminalID>
                  <ipgapi:ProcessorResponseCode>
                        00
                  </ipgapi:ProcessorResponseCode>
                  <ipgapi:ProcessorApprovalCode>
                        440368
                  </ipgapi:ProcessorApprovalCode>
                  <ipgapi:ProcessorReceiptNumber>
                        4291
                  </ipgapi:ProcessorReceiptNumber>
                  <ipgapi:ProcessorTraceNumber>
                        004336
                  </ipgapi:ProcessorTraceNumber>
            </ipgapi:IPGApiOrderResponse>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

11.1.2 Mensagem de falha de SOAP

Em geral, uma mensagem de falha de SOAP devolvida pelo Web Service API tem o seguinte formato:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
            xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
      <SOAP-ENV:Header />
      <SOAP-ENV:Body>
            <SOAP-ENV:Fault>
                  <faultcode>SOAP-ENV:Client</faultcode>
                  <faultstring xml:lang="en-US">
                        <!-- fault message -->
                   </faultstring>
                   <detail>
                        <!-- fault message -->
                   </detail>
             </SOAP-ENV:Fault>
       </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Resumidamente, o elemento `faultstring` carrega o tipo de falha. De acordo com o tipo de falha, os outros elementos são definidos. Observe que nem todos os elementos mostrados precisam ocorrer no elemento `SOAP-ENV:Fault`. A correspondência de elementos para tipo de falha é descrita nas próximas seções.

12.1 SOAP-ENV:Server

Em geral, este tipo de falha indica que o Web Service falhou ao processar sua transação devido a um erro interno do sistema. Se você receber isso como resposta, entre em contato com a equipe de suporte para resolver o problema.

Uma *InternalException* terá sempre a aparência a seguir:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
            xmlns:SOAP-
            ENV="http://schemas.xmlsoap.org/soap/envelope/">
      <SOAP-ENV:Header />
      <SOAP-ENV:Body>
            <SOAP-ENV:Fault>
                  <faultcode>SOAP-ENV:Server</faultcode>
                  <faultstring xml:lang="en-
                        US"> unexpected error
                  </faultstring>
            </SOAP-ENV:Fault>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Os elementos da mensagem de falha de SOAP (relacionados ao elemento `SOAP-ENV:Envelope/SOAP-ENV:Body/SOAP-ENV:Fault`) são definidos da seguinte forma:

Caminho/Nome	Tipo de esquema XML	Descrição
`faultcode`	`xs:string`	Este elemento é sempre definido como `SOAP-ENV:Server`, indicando que a causa do erro é uma falha do sistema subjacente à API.
`faultstring`	`xs:string`	Este elemento sempre carrega a seguinte string com falha: `erro inesperado`

12.2 SOAP-ENV:Client

1. MerchantException

Este tipo de falha ocorrer se o Sicredi e-commerce pode rastrear o erro até sua loja, caso você tenha passado a informação incorreta. Pode haver vários motivos:

a. Sua loja está registrada como sendo fechada. Caso você recebe essa informação apesar de a sua loja esta registrada como aberta, entre em contato com o suporte.
b. A comSicrediação de ID da Loja/ID de Usuário que você informou para autorização de HTTPS estão incorretas sintaticamente.
c. O XML não corresponde ao esquema.

Uma *MerchantException* sempre terá esta aparência:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
            xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
      <SOAP-ENV:Header />
      <SOAP-ENV:Body>
            <SOAP-ENV:Fault>
                  <faultcode>SOAP-ENV:Client</faultcode>
                  <faultstring xml:lang="en-
                        US"> MerchantException
                  </faultstring>
                  <detail>
                        <!-- detailed explanation. -->
                  </detail>
            </SOAP-ENV:Fault>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Os elementos da mensagem de falha de SOAP (relacionados ao elemento `SOAP-ENV:Envelope/SOAP-ENV:Body/SOAP-ENV:Fault`) são definidos da seguinte forma:

Caminho/Nome	Tipo de esquema XML	Descrição
`faultcode`	`xs:string`	Este elemento é sempre definido como `SOAP-ENV:Client`

faultstring	xs:string	Este elemento é sempre definido como `MerchantException`
detail/reason	xs:string	Ao menos um motivo

Consulte a seção Exceções de estabelecimento no Anexo para análises detalhadas de erros.

12.3 ProcessingException

Uma falha deste tipo é levantada sempre que o Sicredi e-commerce tenha detectado um erro durante o processamento da sua transação. A diferença de outros tipos de falha é que a transação passou pela verificação em relação ao xsd.

Uma *ProcessingException* sempre terá esta aparência:

```
<SOAP-ENV:Envelope
            xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
      <SOAP-ENV:Header />
      <SOAP-ENV:Body>
            <SOAP-ENV:Fault>
                  <faultcode>SOAP-ENV:Client</faultcode>
                  <faultstring xml:lang="en-US">
                        ProcessingException: Processing the request
                        resulted in an error - see SOAP details for
                        more information
                  </faultstring>
                  <detail>
                        <ipgapi:IPGApiOrderResponse
            xmlns:ipgapi="https://ipg-online.com/ipgapi/schemes/ipgapi">
                              <ipgapi:CommercialServiceProvider
                                    > BNLP
                              </ipgapi:CommercialServiceProvider>
                              <ipgapi:TransactionTime>
                                    1192111156423
                              </ipgapi:TransactionTime>
                              <ipgapi:ProcessorReferenceNumber />
                              <ipgapi:ProcessorResponseMessage>
                                    Card expiry date exceeded
                              </ipgapi:ProcessorResponseMessage>
                              <ipgapi:ErrorMessage>
                                    SGS-000033: Card expiry date exceeded
                              </ipgapi:ErrorMessage>
                              <ipgapi:OrderId>
                                    62e3b5df-2911-4e89-8356-
                                    1e49302b1807
                              </ipgapi:OrderId>
                              <ipgapi:ApprovalCode />
                              <ipgapi:AVSResponse />
                              <ipgapi:TDate>1192139943</ipgapi:TDate>
                              <ipgapi:TransactionResult>
                                    FAILED
                              </ipgapi:TransactionResult>
                              <ipgapi:TerminalID>123456</ipgapi:TerminalI
                              D>
                              <ipgapi:ProcessorResponseCode/>
                              <ipgapi:ProcessorApprovalCode />
                              <ipgapi:ProcessorReceiptNumber />
                              <ipgapi:ProcessorTraceNumber />
                        </ipgapi:IPGApiOrderResponse>
                  </detail>
            </SOAP-ENV:Fault>
      </SOAP-ENV:Body>
```

```
</SOAP-ENV:Envelope>
```

Os elementos da mensagem de falha de SOAP (relacionados ao elemento `SOAP-ENV:Envelope/SOAP-ENV:Body/SOAP-ENV:Fault`) são definidos da seguinte forma:

Caminho/Nome	Tipo de esquema XML	Descrição
`faultcode`	`xs:string`	Este elemento é sempre definido como `SOAP-ENV:Client`, indicando que a causa do erro foi provavelmente encontrada em dados de transação inválidos que foram passados.
`faultstring`	`xs:string`	Este elemento sempre carrega a seguinte string com falha: `ProcessingException`
`detail/ ipgapi:IPGApiOrderResponse`	Elemento composto	Este elemento contém o erro. Como há diversas causas para levantar esse tipo de exceção, o próximo capítulo apresentará uma visão geral, explicando os dados contidos neste elemento.

Consulte a seção Como processar exceções no Anexo para análises detalhadas de erros.

13. Como analisar o resultado da transação

13.1 Aprovação de transação

A mensagem de SOAP envolvendo a aprovação de uma transação foi apresentada no capítulo anterior, juntamente com um exemplo. O relatório de status da transação é contido no elemento `ipgapi:IPGApiOrderResponse` e pode ser entendido como os dados retornados pela operação do Web Service. A seguir, seus elementos (relativo ao super elemento `ipgapi:IPGApiOrderResponse`) conforme descrito. Observe que o conjunto completo de elementos está sempre presente na resposta. Contudo, alguns elementos podem estar vazios.

Caminho/Nome	Tipo de esquema XML	Descrição
`ipgapi: CommercialServiceProvider`	`xs:string`	Indica seu provedor.
`ipgapi:TransactionTime`	`xs:string`	O carimbo de data/hora é definido pelo Sicredi e-commerce antes de devolver a aprovação da transação.
`ipgapi: ProcessorReferenceNumber`	`xs:string`	Em alguns casos, este elemento pode estar vazio. Ele armazena um número que permite que o processador de cartão de crédito faça referência a essa transação. Você não precisa informar esse número em transações futuras. Contudo, tenha este número à mão caso você perceba algum problema com a transação e precise entrar em contato com o suporte.

ipgapi: ProcessorResponseMessage	xs:string	Em caso de aprovação, este elemento contém a string: `Função executada sem erros (Function performed error-free)`
ipgapi: ProcessorResponseCode	xs:string	O código de resposta do processador de cartão de crédito
ipgapi:ErrorMessage	xs:string	Este elemento fica vazio em caso de uma aprovação.
ipgapi:OrderId	xs:string	Este elemento contém o Número do Pedido (Order ID).Para transações de *Venda (Sale)* e *PreAuth*, um novo Order ID é informado. Para transações de *PostAuth*, *Cancelamento* e *Estorno*, informe esse número no elemento `v1:OrderId` para esclarecer a qual transação você se refere. O elemento `ipgapi:OrderId` de uma aprovação de transação para uma transação de *PostAuth*, *Cancelamento* ou *Estorno* apenas retorna o Order ID (Número do Pedido) ao qual a transação se refere.
ipgapi:ApprovalCode	xs:string	Armazena o código de autorização que o processador de transação criou para essa transação. Você não precisa informar esse código em transações futuras. Contudo, tenha este número à mão caso você perceba algum problema com a transação e precise entrar em contato com o suporte.
ipgapi:AVSResponse	xs:string	Devolve a resposta do sistema de verificação de endereços (address verification system, AVS) – Inicialmente não oferecido para o Brasil pela Sicredi.
ipgapi:TDate	xs:string	Armazena a TDate que você tem que fornecer para estornar (VOID) essa transação (o que é possível somente para transações de *Venda (Sale)* e *PostAuth (Captura Posterior)*). Neste caso, passe seu valor no elemento `v1:TDate` da transação de *Estorno (VOID)* que você deseja criar.
ipgapi:TransactionResult	xs:string	Armazena o resultado de transação que é sempre definido como `APPROVED` no caso de uma aprovação.
ipgapi:TerminalID	xs:string	O ID do Terminal usado para esta transação.
ipgapi:PaymentType	xs:string	O tipo de pagamento usado para esta transação.
ipgapi:Brand	xs:string	A bandeira do cartão usada para esta transação.
ipgapi:Country	xs:string	O país de emissão do cartão que foi usado para esta transação.

13.2 Falha na transação

Como mostrado no capítulo anterior, uma mensagem de falha de SOAP, resultante da falha no processamento da sua transação, contém um elemento `ipgapi:IPGApiOrderResponse` passado como secundário de um elemento de detalhe de SOAP. Observe que seus subelementos são exatamente os mesmos que os de uma aprovação da transação. O seu significado no caso de falha é descrito abaixo:

Caminho/Nome	Tipo de esquema XML	Descrição
`ipgapi: CommercialServiceProvider`	`xs:string`	Indica seu provedor.
`ipgapi:TransactionTime`	`xs:string`	O carimbo de data/hora é definido pelo Sicredi e-commerce antes de devolver a falha da transação.
`ipgapi: ProcessorReferenceNumber`	`xs:string`	Em alguns casos, este elemento pode estar vazio. Armazena um número que permite que o processador de cartão de crédito faça referência a essa transação. Você não precisa informar esse número em transações futuras. Contudo, tenha este número à mão caso você perceba algum problema com a transação e precise entrar em contato com o suporte.
`ipgapi: ProcessorResponseMessage`	`xs:string`	Armazena a mensagem de erro devolvida pelo processador de cartão de crédito. Por exemplo, no caso de um cartão de crédito vencido, isso poderia ser: `Data de validade do cartão vencida`
`ipgapi: ProcessorResponseCode`	`xs:string`	O código de resposta do processador de cartão de crédito
`ipgapi: ProcessorApprovalCode`	`xs:string`	O código de autorização do processador de cartão de crédito
`ipgapi: ProcessorReceiptNumber`	`xs:string`	O número de comprovante de pagamento do processador de cartão de crédito
`ipgapi: ProcessorTraceNumber`	`xs:string`	O número de rastreio do processador de cartão de crédito
`ipgapi:ErrorMessage`	`xs:string`	Armazena a mensagem de erro devolvida pelo Sicredi e-commerce. Ela está sempre codificada no formato *SGS-XXXXXX:* Uma *mensagem* com *XXXXXX* sendo um código de erro de seis dígitos e uma *mensagem* descrevendo o erro (essa descrição pode ser diferente da mensagem de resposta do processador). Por exemplo, no exemplo acima, a mensagem de erro `SGS-000033: Data de validade do cartão vencida` é devolvida. Tenha o código de erro e a mensagem prontas ao entrar em contato com o suporte.

ipgapi:OrderId	xs:string	Armazena o Número do Pedido (Order ID). Ao contrário de uma aprovação, este Order ID nunca é exigido para futuras transações, mas é necessário para rastrear a causa do erro. Portanto, tenha-o pronto ao entrar em contato com o suporte.
ipgapi:ApprovalCode	xs:string	Este elemento fica vazio em caso de uma falha na transação.
ipgapi:AVSResponse	xs:string	Devolve a resposta do sistema de verificação de endereços (address verification system, AVS) – Inicialmente não oferecido para o Brasil pela Sicredi.
ipgapi:TDate	xs:string	Armazena a TDate. Assim como ocorre com o Order ID (Número do Pedido), a TDate nunca é exigida para futuras transações, mas é necessária para rastrear a causa do erro. Portanto, tenha-o pronto ao entrar em contato com o suporte.
ipgapi:TransactionResult	xs:string	Em caso de falha, há três valores possíveis: • DECLINED • FRAUD • FAILED DECLINED é devolvido caso o processador de cartão de crédito não aceite a transação, por exemplo, quando o limite do cartão do cliente não é suficiente. FRAUD é devolvido no caso de uma tentativa de fraude que é presumida pelo Sicredi e- commerce. Se houver um erro interno, o valor retornado é FAILED.
ipgapi:TerminalID	xs:string	O ID do Terminal usado para esta transação.

14. Como criar uma solicitação POST de HTTPS

A criação de uma solicitação POST de HTTPS é uma tarefa que raramente precisa ser feita de forma manual. Há várias ferramentas e bibliotecas para apoio na criação de solicitações de HTTPS. Em geral, a funcionalidade exigida para realizar esta tarefa está presente no conjunto padrão de bibliotecas que acompanha o ambiente tecnológico no qual você desenvolveu sua loja online.
Como todas essas bibliotecas diferem um pouco com relação ao seu uso, não há um processo de criação geral que possa ser descrito. A fim de ilustrar os conceitos básicos, os capítulos a seguir dão exemplos de como criar uma solicitação de HTTPS em PHP e ASP.
Em geral, o conjunto de parâmetros que você precisa informar na criação de uma solicitação de HTTPS válida em qualquer tecnologia é o seguinte:

Parâmetro	Valor	Descrição
URL	`https://test.ipg-online.com/ipgapi/services`	Este é o URL completo do Web Service API; dependendo da funcionalidade que você usa para criar solicitações de HTTP, você talvez precise dividir esse URL entre o host e o service e informar isso nos cabeçalhos apropriados de solicitação de HTTP.
Content-Type	`text/xml`	Este é um cabeçalho de HTTP adicional que precisa ser definido. Isso ocorre porque a mensagem de solicitação SOAP é codificada em XML e passada como conteúdo no corpo da solicitação POST de HTTP.
Autorização	Tipo: `Básico` Nome de usuário: `WS`*storeID*`._.`*userID* Senha: *yourPassword*	Sua loja é identificada no Sicredi e-commerce na verificação dessas credenciais. A fim de usar o Web Service API, você precisa fornecer seu ID da Loja, ID de Usuário e senha como conteúdo de um cabeçalho de autorização *Básico* de HTTP. Por exemplo, digamos que seu ID da Loja é `101`, seu ID de Usuário é `007` e sua senha é `myPW`. O nome do usuário de autorização é `WS101._.007`. O cabeçalho completo de autorização de HTTP deve ser: `Autorização: Básico V1MxMDEuXy4wMDc6bXlQVw==` Observe que a última string é o resultado da codificação base 64 da string `WS101._.007:myPW`.
Corpo de HTTP	XML de solicitação de SOAP	O corpo de solicitação POST de HTTP assume a mensagem de solicitação SOAP

15. PHP

Fazer a comunicação HTTP em PHP é sobretudo realizada com o auxílio do cURL que é enviado tanto como uma biblioteca quanto como uma ferramenta de linha de comando. Em versões mais recentes de PHP, o cURL já é incluído como uma extensão que precisa ser "ativada". Por isso, a funcionalidade do cURL está disponível em qualquer script de PHP. Apesar de essa ser uma tarefa objetiva, caso seu Web Service opere no Microsoft Windows, ela pode precisar da compilação de PHP em máquinas com Unix/Linux. Portanto, você deve considerar chamar a ferramenta de linha de comando de cURL pelo seu script PHP em vez de usar a extensão cURL. As duas variantes são consideradas no início a seguir com o uso da extensão do cURL no PHP 5.2.4 executado em uma máquina Windows.

15.1 Como usar a extensão PHP de cURL

No geral, a ativação da extensão do cURL no PHP 5.2.4 apenas exige *"descomentar"* (uncomment) a seguinte linha no seu arquivo de configuração *php.ini*:

```
;extension=php_curl.dll
```

Observe que outras versões de PHP podem exigir outras ações para ativar a compatibilidade do cURL no PHP. Consulte a documentação de PHP para obter mais informações. Depois de ativar o cURL, uma solicitação de HTTP com os parâmetros acima é definida com as seguintes instruções de PHP:

```
<?php
// storing the SOAP message in a variable – note that the plain XML code
// is passed here as string for reasons of simplicity, however, it is
// certainly a good practice to build the XML e.g. with DOM – furthermore,
// when using special characters, you should make sure that the XML string
// gets UTF-8 encoded (which is not done here):
$body = "<SOAP-ENV:Envelope ...>...</SOAP-ENV:Envelope>";
// initializing cURL with the IPG API URL:
$ch = curl_init("https://test.ipg-online.com/ipgapi/services");
// setting the request type to POST:
curl_setopt($ch, CURLOPT_POST, 1);
// setting the content type:
curl_setopt($ch, CURLOPT_HTTPHEADER, array("Content-Type: text/xml"));
// setting the authorization method to BASIC:
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
// supplying your credentials:
curl_setopt($ch, CURLOPT_USERPWD, "WS101._.007:myPW");
// filling the request body with your SOAP message:
curl_setopt($ch, CURLOPT_POSTFIELDS, $body);
...
?>
```

A definição das opções de segurança que são necessárias para habilitar a comunicação de SSL será discutida no próximo capítulo, ampliando o script acima.

15.2 Como usar a ferramenta de linha de comando cURL

Pelos motivos descritos acima, você deve considerar o uso da ferramenta de linha de comando cURL em vez da extensão. O uso da ferramenta não exige nenhum esforço de configuração de PHP: seu script de PHP apenas precisa chamar o executável com um conjunto de parâmetros. Como as configurações de segurança somente serão abordadas no próximo capítulo, o script a seguir apenas mostra como configurar os parâmetros de HTTP padrão, ou seja, o script é ampliado com os parâmetros de SSL no próximo capítulo.

```
<?php
// storing the SOAP message in a variable – note that you have to escape
// " and \n, since the latter makes the command line tool fail,
// furthermore note that the plain XML code is passed here as string
// for reasons of simplicity, however, it is certainly a good practice
// to build the XML e.g. with DOM – finally, when using special
// characters, you should make sure that the XML string gets UTF-8 encoded
// (which is not done here):
$body = "<SOAP-ENV:Envelope ...>...</SOAP-ENV:Envelope>";
// setting the path to the cURL command line tool – adapt this path to the
// path where you have saved the cURL Sicrediaries:
$path = "C:\curl\curl.exe";
// setting the IPG API URL:
$apiUrl = " https://test.ipg-online.com/ipgapi/services";
// setting the content type:
$contentType = " --header \"Content-Type: text/xml\"";
// setting the authorization method to BASIC and supplying
// your credentials:
$user = " --basic --user WS101._.007:myPW";
// setting the request body with your SOAP message – this automatically
// marks the request as POST:
$data = " --data \"".$body."\"".
...
?>
```

15.3 ASP

Há várias formas de criar uma solicitação de HTTP em ASP. Entretanto, a seguir, o uso do WinHTTP 5.1 é descrito em conjunto com o Windows Server 2003 e o Windows XP SP2. Além disso, apenas algumas linhas de código são necessárias para configurar uma solicitação de HTTP válida. Observe que o fragmento de código a seguir foi escrito em JavaScript. O uso do VB Script no lugar JavaScript não altera a base das instruções apresentadas.

```
<%@ language="javascript"%>
<html>...<body>
<%
// storing the SOAP message in a variable – note that the plain XML code
// is passed here as string for reasons of simplicity, however, it is
// certainly a good practice to build the XML e.g. with DOM – furthermore,
// when using special characters, you should make sure that the XML string
// gets UTF-8 encoded (which is not done here):
var body = "<SOAP-ENV:Envelope ...>...</SOAP-ENV:Envelope>";
// constructing the request object:
var request = Server.createObject("WinHttp.WinHttpRequest.5.1");
// initializing the request object with the HTTP method POST
// and the IPG API URL:
request.open("POST", "https://test.ipg-online.com/ipgapi/services");
// setting the content type:
request.setRequestHeader("Content-Type", "text/xml");
// setting the credentials:
request.setCredentials("WS10036000750._.1001", "testinger", 0);
...
%>
</body></html>
```

Observe que o script acima é ampliado no próximo capítulo ao definir as opções de segurança que são necessárias para estabelecer o canal SSL.

16. Como estabelecer uma conexão TLS

Antes de enviar a solicitação de HTTP criada no capítulo anterior, um canal de comunicação segura precisa ser estabelecido, garantindo que todo os dados sejam passados de forma criptografada e que o cliente (sua aplicação) e o servidor (que executa a Web Service API) tenham certeza que estão se comunicando um com o outro.

É possível fazer isso ao estabelecer uma conexão TLS com os certificados de troca de cliente (client) e de servidor (Server). Um certificado identifica um dos lados da comunicação de forma exclusiva. Basicamente, o processo funciona assim:

1. TLS: o cliente solicita acesso a www.ipg-online.com
2. TLS: o servidor apresenta seu certificado ao cliente
3. TLS: o cliente verifica o certificado do servidor (opcional)
4. TLS: o servidor solicita ao cliente um certificado de cliente
5. TLS: o cliente envia seu certificado para o servidor
6. TLS: o servidor verifica as credenciais do cliente
7. TLS: se bem-sucedido, o servidor estabelece túnel TLS em www.ipg-online.com e todos os dados trocados entre as partes são criptografados.
8. HTTP: Inicie o HTTP e solicite a parte do URL: / ipgapi / services [...]

Depois desse processo, o aplicativo precisa fazer duas coisas: Primeiro, iniciar a comunicação ao enviar seu certificado de cliente. Segundo, verificar o certificado do servidor recebido. A forma como isso é feito varia de plataforma para plataforma. No entanto, para ilustrar os conceitos básicos, os scripts PHP e ASP iniciados no capítulo anterior serão continuados ao ampliá-los com as instruções relevantes necessárias para a configuração de uma conexão TLS.

16.1 PHP

Retomando a distinção entre usar a extensão cURL de PHP ou a ferramenta de linha de comando, as duas seções a seguir continuarão a abordar as duas maneiras diferentes de ativar a comunicação segura de HTTP. No entanto, independentemente da sua abordagem, você será confrontado com uma característica especial do cURL: ele requer que o certificado de cliente seja transmitido como arquivo PEM com a chave privada do certificado do cliente transmitida em um arquivo extra. Por fim, a senha da chave privada de certificado de cliente precisa ser informada. De forma simples, o arquivo PEM contém o certificado com todas as informações necessárias para permitir que o servidor identifique o cliente. A chave privada não é realmente necessária para esse tipo de comunicação. No entanto, é crucial para o cURL funcionar.

16.2 Como usar a extensão PHP cURL

Após a criação do script inicial no capítulo anterior, os parâmetros que são necessários para estabelecer uma conexão SSL com cURL são definidos nas seguintes demonstrações:

```
<?php
...
// telling cURL to verify the server certificate:
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 1);
// setting the path where cURL can find the certificate to verify the
// received server certificate against:
curl_setopt($ch, CURLOPT_CAINFO, "C:\certs\geotrust.pem");
// setting the path where cURL can find the client certificate:
curl_setopt($ch, CURLOPT_SSLCERT, "C:\certs\WS101._.007.pem");
// setting the path where cURL can find the client certificate's
// private key:
curl_setopt($ch, CURLOPT_SSLKEY, "C:\certs\WS101._.007.key");
// setting the key password:
curl_setopt($ch, CURLOPT_SSLKEYPASSWD, "ckp_1193927132");
...
?>
```

Observe que este script é ampliado no próximo capítulo pelas instruções que fazem a solicitação HTTP real.

16.2.1 Como usar a ferramenta de linha de comando cURL

Após a criação do script inicial no capítulo anterior, as instruções que inicializam os parâmetros TLS transmitidos para a ferramenta de linha de comando cURL são as seguintes:

```
<?php
...
// setting the path where cURL can find the certificate to verify the
// received server certificate against:
$serverCert = " --cacert C:\certs\geotrust.pem";
// setting the path where cURL can find the client certificate:
$clientCert = " --cert C:\certs\WS101._.007.pem";
// setting the path where cURL can find the client certificate's
// private key:
$clientKey = " --key C:\certs\WS101._.007.key";
// setting the key password:
$keyPW = " --pass ckp_1193927132";
...
?>
```

Observe que este script é ampliado no próximo capítulo pelas instruções que fazem a solicitação HTTP real.

16.3 ASP

Para fazer com que o processo de inicialização TLS acima funcione, o ASP exige tanto o certificado de cliente como o certificado do servidor nos armazenamentos de certificado. Em outras palavras, antes de o ASP comunicar-se via TLS, ambos os certificados têm de ser instalados pela primeira vez. As etapas seguintes, que presumem a execução do ASP no Microsoft IIS 5.1 no Windows XP, o guiarão pelo processo de configuração:

1. Clique em *Iniciar*, clique em *Executar...*, digite *mmc* e clique em *OK*.
2. Abra o menu *Arquivo* e selecione *Adicionar/remover snap-in*.
3. Clique em *Adicionar*.
4. Em *Snap-In*, escolha *Certificados* e clique em *Adicionar*.
5. Você deverá selecionar a conta para a qual deseja gerenciar os certificados. Como o IIS usa a conta do computador, escolha *Conta de computador* e clique em *Avançar*.
6. Escolha *Computador Local* e clique em *Terminar*.
7. Clique em *Fechar* e em *OK*.
8. Expanda a árvore de *Certificados (Computador Local)*. O certificado de cliente será instalado na pasta *Pessoal*.
9. Em seguida, clique com o botão direito na pasta *Certificados*, selecione *Todas as Tarefas* e clique em *Importar...*. Isso abrirá o Assistente para Importação de Certificados.
10. Clique em *Avançar*. Escolhe seu arquivo p12 do certificado de cliente e clique em *Avançar*.
11. Informe a senha de instalação do certificado de cliente e clique em *Avançar*.
12. Selecione *Colocar todos os certificados no armazenamento a seguir* e procure pela pasta*Pessoal*, se ela não for exibida. Clique em *Avançar*.
13. Verifique as configurações exibidas e clique em *Terminar*. Seu certificado de cliente agora está instalado no armazenamento de certificados pessoais do computador local. Aqui, o IIS (executando o ASP) pode consultar o certificado de cliente ao se comunicar com outro servidor via HTTP.
14. Agora, o certificado do servidor tem de ser instalado no armazenamento *Autoridades de Certificação Confiáveis*. Os certificados neste armazenamento são usados para verificação sempre que recebem um certificado de um servidor. Isso significa que o certificado do servidor da Web Service API deve ser instalado aqui. Dessa forma, o IIS é capaz de verificar o certificado do servidor recebido quando entrar em contato com o Serviço Web. Escolha *Autoridades de Certificação Confiáveis* na árvore *Certificados (Computador Local)* e abra a subpasta *Certificados*.
15. Clique com o botão direito na pasta *Certificados*, selecione *Todas as Tarefas* e clique em *Importar...*. Isso abrirá o Assistente para Importação de Certificados.
16. Clique em *Avançar*. Escolha o arquivo PEM do certificado do servidor e clique em *Avançar*.
17. Selecione *Colocar todos os certificados no armazenamento a seguir* e procure pela pasta*Autoridades de Certificação Confiáveis*, se ela já não estiver em exibição. Clique em *Avançar*.
18. Verifique as configurações exibidas e clique em *Terminar*. O certificado do servidor agora está instalado no armazenamento de certificados seguros do computador local. Aqui, o IIS pode consultar o certificado do servidor para verificação em relação ao certificado do servidor da Web Service API recebido durante o processo de configuração do TLS.

Depois de instalar os dois certificados, é possível imaginar que o ambiente que permite a comunicação de ASP via TLS está definido. Contudo, ainda há um detalhe que não permite a comunicação: O IIS, que executa seu ASP, tem um usuário Windows que não tem os direitos necessários de acesso à chave privada de certificado de cliente. Apesar de o acesso à chave privada não ser realmente necessário para estabelecer a conexão TLS ao Sicredi e-commerce, o usuário do IIS precisa de direitos de acesso para executar o processo de autenticação no ASP. Para conceder direitos a um usuário, a Microsoft oferece a ferramenta *WinHttpCertCfg.exe*, que você pode baixar gratuitamente em:

http://www.microsoft.com/downloads/details.aspx?familyid=c42e27ac-3409-40e9-8667-c748e422833f&displaylang=en

Depois de instalar a ferramenta, abra o prompt de comando, alterne para o diretório onde você instalou a ferramenta e digite esta linha para conceder o acesso ao usuário do ISS:

```
winhttpcertcfg -g -c LOCAL_MACHINE\My -s WS101._.007 -a IWAM_MyMachine
```

`LOCAL_MACHINE\My` determina o armazenamento chave onde os certificados pessoais da contada máquina local são armazenados. Depois de instalar o certificado de cliente no armazenamento de certificados pessoais conforme descrito acima, o certificado de cliente pode ser encontrado neste caminho, então não há necessidade de informar outro caminho. `WS101._.007` é o nome do certificado de cliente. Você precisa adaptar esse nome ao nome do seu certificado de cliente. Portanto, verifique o nome exibido para o certificado de cliente no console *mmc* depois de instalá-lo conforme descrito acima. Por fim, `IWAM_MyMachine` determina o nome de usuário do IIS. Observe que o IIS 5.1 usa `IWAM_`*MachineName* por padrão. Isso significa que se o nome da sua máquina é *IISServerMachine*, o usuário do IIS chama-se `IWAM_IISServerMachine`. Observe que outras versões do IIS talvez usem um esquema de nomes diferente. Se você não sabe o nome da sua máquina ou o nome de usuário do ISS, verifique a documentação do IIS e entre em contato com o administrador.

Agora você está pronto para usar TLS no seu código ASP. O código que amplia o script de ASP iniciado no capítulo anterior é reduzido a apenas uma instrução adicional que diz ao WinHTTP para qual certificado de cliente enviar (e onde encontrá-lo) ao entrar em contato com o Sicredi e-commerce:

```
<%@ language="javascript"%>
<html>...<body>
<%
...
// setting the path where the client certificate to send can be found:
request.setClientCertificate("LOCAL_MACHINE\\My\\WS101._.007");
...
%>
</body></html>
```

Observe que se você usa VB Script, o código parece quase o mesmo; contudo, não esqueça de substituir as barras invertidas duplas no caminho por uma barra (ou seja, o caminho para o certificado deveria ser `"LOCAL_MACHINE\My\WS101._.007"`).

Observe que este script é ampliado no próximo capítulo pelas instruções que fazem a solicitação HTTP real.

17. Como enviar a solicitação POST de HTTPS e receber a resposta

A comunicação real com o Web Service API ocorre ao enviar a solicitação de HTTPS e aguardar uma resposta. Novamente, como isso é feito depende da tecnologia que você está usando. A maioria das bibliotecas de HTTP cobre os detalhes de comunicação subjacentes e reduzem esse processo a uma única chamada de operação ao devolver a resposta de HTTP como resultado.
De qualquer forma, os parâmetros que são exigidos para realizar com sucesso uma solicitação POST de HTTP sobre SSL e receber a resposta (carregar um código de status de HTTP `200`) foram descritos nos dois capítulos anteriores. Ao definir parâmetros inválidos ou incorretos, o servidor web executando a Web Service API devolve um código de erro HTTP padrão no cabeçalho de HTTP da resposta ou envia uma falha de SSL. Seu significado pode ser encontrado em qualquer guia de HTTP/SSL.

Contudo, há uma exceção importante: No caso dos parâmetros de HTTP que você informou estarem corretos, mas o Web Service falhou ao processar sua transação devido a um valor incorreto contido na mensagem de solicitação SOAP (por exemplo, um número de cartão de crédito inválido), uma exceção de SOAP é lançada e transferida no corpo de uma resposta de HTTP carregando o código de erro `500`. Os detalhes sobre a causa da exceção são informados na mensagem de falha de SOAP descrita no contexto do próximo capítulo.

A fim de concluir os scripts de PHP e ASP, criados gradualmente nos capítulos anteriores, os dois capítulos a seguir fornecerão as instruções necessárias para fazer um chamado de HTTP usando essas tecnologias.

17.1 PHP

Novamente, a distinção entre a extensão PHP cURL e a ferramenta de linha de comando do cURL é feita a seguir:

17.1.1 Como usar a extensão PHP cURL

O script PHP usando a extensão do cURL é finalmente concluído ao chamar com as instruções mostradas abaixo. Observe que a chamada de HTTP retorna uma resposta de SOAP ou mensagem de falha no corpo da resposta de HTTP.

```
<?php
...
// telling cURL to return the HTTP response body as operation result
// value when calling curl_exec:
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
// calling cURL and saving the SOAP response message in a variable which
// contains a string like "<SOAP-ENV:Envelope ...>...</SOAP-ENV:Envelope>":
$result = curl_exec($ch);
// closing cURL:
curl_close($ch);
?>
```

17.1.2 Como usar a ferramenta de linha de comando cURL

Chamar o HTTP com a ferramenta de linha de comando cURL exige a conclusão da instrução de linha de comando e execução da ferramenta externa. Contudo, a leitura da resposta de HTTP é mais complicada porque o comando `exec` de PHP salva cada linha retornada por um programa externo como um elemento de uma matriz. Ao concatenar todos os elementos da matriz, obtém-se a resposta de SOAP ou mensagem de falha que foi devolvida no corpo de resposta de HTTP. As instruções a seguir abordam o chamado de HTTP e a conclusão do script:

```
<?php
...
// saving the whole command in one variable:
$curl = $path.
      $data.
      $contentType.
      $user.
      $serverCert.
      $clientCert.
      $clientKey.
      $keyPW.
      $apiUrl;
// preparing the array containing the lines returned by the cURL
// command line tool:
$returnArray = array();
// performing the HTTP call by executing the cURL command line tool:
exec($curl, $returnArray);
// preparing a variable taking the complete result:
$result = "";
// concatenating the different lines returned by the cURL command
// line tool – this result in the variable $result carrying the entire
// SOAP response message as string:
foreach($returnArray as $item)
      $result = $result.$item;
?>
```

17.1.3 ASP

A realização da chamada de HTTP com WinHTTP em ASP é limitada a uma única chamada de operação que usa o XML da solicitação de SOAP como parâmetro. Depois de realizar a solicitação, a resposta de SOAP ou a mensagem de falha é devolvida, o que pode ser recuperado no formato de uma string ao acessar a propriedade `responseText` do objeto da solicitação. A aparência da mensagem de resposta SOAP é descrita no próximo capítulo. As instruções a seguir concluem o script de ASP:

```
 <%@ language="javascript"%>
<html>...<body>
<%
...
// doing the HTTP call with the SOAP request message as input:
request.send(body);
// saving the SOAP response message in a string variable:
var response = request.responseText;
%>
</body></html>
```

18. Como usar o Java client para conectar-se ao Web Service

Para uma integração rápida e simples, a SICREDI oferece um Java client para conectar- se ao Web Service do Sicredi e-commerce. Uma instância da classe IPGApiClient gerencia a conexão ao web service, cria o XML e as mensagens de SOAP e avalia as respostas. Para criar uma transação ou administrar uma resposta, o desenvolvedor trabalha com classes simples de Java bean.

O IPGApiClient usa o apache http client. Algumas configurações do http client afetam qualquer http client para o mesmo ambiente de class loader.

18.1 Criar uma instância de IPGApiClient

Há vários construtores disponíveis para criar instâncias de IPGApiClient. O exemplo abaixo ilustra como usar o mais simples deles. O método getBytes também é incluído para a conclusão e simplificação do exemplo.

```
String url = "https://test.ipg-online.com/ipgapi/services";
String storeId = "your store id";
String password = "your password";
byte[] key = getBytes("/path/to/your/keyStore.ks");
String keyPW = "your key store password";
IPGApiClient client = new IPGApiClient(url, storeId, password, key, keyPW);
/**
 * getBytes
 * reads a resource and returns a byte array
 * @param resource the resource to read
 * @return the resource as byte array
 */
public static byte[] getBytes(final String resource) throws IOException {
      final InputStream input = IO.class.getResourceAsStream(resource);
      if (input == null) {
            throw new IOException(resource);
                                    }
      try {
final byte[] bytes = new byte[input.available()];
input.read(bytes);
return bytes;
                                                                  } finally {
                                                                        try {
input.close();
            } catch (IOException e) {
                  log.warn(resource);
            }
      }
}
```

18.2 Como construir uma transação e administrar a resposta

Há diferentes classes para transações com tipos de cartão. Segue exemplo:

- Cartão de crédito

A classe de família a seguir pode ser usada para gerar a classe necessária:

```
de.firstdata.ipgapi.client.transaction.IPGApiTransactionFactory
```

O exemplo abaixo mostra uma transação de Venda à vista de cartão de crédito para um valor de 7 euros:

```
Amount amount = new Amount("7", "978"); // ISO 4217: EUR = 978
CreditCard cC = new CreditCard("1111222233334444", "07", "17", null);
CCSaleTransaction transaction =
      IPGApiTransactionFactory.createSaleTransactionCredit(amount, cC);
// some transactions may include further information e.g. the customer
transaction.setName("a name");
try {
      IPGApiResult result = client.commitTransaction(transaction);
      // now you can read the conclusion
      System.out.println(result.getOrderId());
      System.out.println(result.getTransactionTime());
      // ...
} catch (ProcessingException e) {
      // ERROR: transaction not passed
}
```

18.3 Como construir uma ação

A classe de família a seguir pode ser usada para gerar a classe necessária:

```
de.firstdata.ipgapi.client.transaction.IPGApiActionFactory
```

Para enviar uma ação, você precisa usar o método *commitAction* do IPGApiClient. O processo a seguir é parecido com transações de pagamento.

18.4 Como conectar-se por trás de um proxy

Antes de usar o IPGApiClient por trás de um proxy, você deve definir a configuração de proxy do cliente com o método IPGApiClient:

```
IPGApiClient.setProxy(
      final String host, final Integer port,
      final String user, final String password,
      final String workstation, final String domain)
```

Os parâmetros user, password, workstation e domain devem ser nulos se não é necessária identificação. Se você precisa se identificar em um proxy de MS Windows, você deve definir o domínio do parâmetro. Para identificar-se em sistema como Unix, o domínio de parâmetro deve ser nulo. Para obter mais informações, consulte o apache javadoc.

Depois de definir os parâmetros de proxy, você deve chamar o método IPGApiClient.init().

19. Usando .NET Client para se conectar ao web service

Da mesma forma que o Java client descrito no capítulo anterior, fornecemos um .NET client para uma integração rápida e simples ao web service. Entre em contato com a equipe de suporte Sicredi para obter o arquivo zip que também contém uma descrição sobre como adicionar a Web Reference e como usar o client.

20. Anexo

XML

A Web Service API usa o padrão de XML para comunicação conforme descrito em

http://www.w3.org/standards/xml/core

, incluindo a especificação de namespaces descrita em

http://www.w3.org/TR/2009/REC-xml-names-20091208/

Para alterar os nomes das tags exclusivas de XML (por ex., no IPG: IPGApiActionRequest, Action, RecurringPayment, etc.), namespaces são usados.

Exemplo:
`http://ipg-online.com/ipgapi/schemas/ipgapi`, `http://ipg-online.com/ipgapi/schemas/a1`, …

Esses namespaces são definidos nos arquivos xsd, como
`xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi"`.

Os mesmos namespaces devem ser declarados nos arquivos XML (sem análise com referências de namespace de código fixo), iniciando com a palavra-chave `xmlns.`

Para evitar erros com os namespaces, recomendamos usar bibliotecas para gerenciar mensagens de XML.

Futuramente no desenvolvimento do produto, talvez seja necessário ampliarmos a IPGApiRequest ou a IPGApiResponse com mais membros. Apesar da extensão da request não impactar o código implementado, a ampliação da resposta pode causar erros se você verificar a resposta em comparação com ipgapi.xsd. Portanto, recomendamos desativar a verificação.

Esquemas XML

As definições para os blocos de construção de XML podem ser encontradas aqui:

ipgapi.xsd	https://www.ipg-online.com/ipgapi/schemas/ipgapi.xsd
v1.xsd	https://www.ipg-online.com/ipgapi/schemas/v1.xsd
a1.xsd	https://www.ipg-online.com/ipgapi/schemas/a1.xsd

20.1 Resolução de problemas – Exceções de estabelecimento

```
<detail>
      XML is not wellformed: Premature end of message.
</detail>
```

Explicação possível:
Você enviou uma mensagem totalmente vazia. A mensagem não contém uma mensagem de SOAP, uma mensagem de API do Sicredi e-commerce ou qualquer caractere no corpo do http.

```
<detail>
      XML is not wellformed: Content is not allowed in prolog.
</detail>
```

Explicação possível:
A mensagem não pode ser interpretada como uma mensagem XML.

```
<detail>
      XML is not wellformed:
      XML document structures must start and end within the same entity.
</detail>
```

Explicação possível:
A mensagem inicia como uma mensagem XML, mas a tag final da primeira tag de abertura está ausente.

```
<detail>
      XML is not wellformed:
      The element type "SOAP-ENV:Body" must be terminated
      by the matching end-tag "&lt;/SOAP-ENV:Body&gt;".
</detail>
```

Explicação possível:
Falta uma tag final para uma tag interna aberta (que não seja um tag de nível superior). Neste exemplo, a tag final </SOAP-ENV:Body> está ausente.

```
<detail>
      XML is not wellformed:
      Element type "irgend" must be followed by either attribute
      specifications, "&gt;" or "/&gt;".
</detail>
```

Explicação possível:
A mensagem não é uma mensagem XML ou uma mensagem XML correta. Um caractere “>” está faltando da tag irgend.

```
<detail>
      XML is not wellformed:
      Open quote is expected for attribute "xmlns:ns3"
      associated with an element type "ns3:IPGApiOrderRequest".
</detail>
```

Explicação possível:
O valor de um atributo não está disposto entre aspas. Na API, somente atributos são usados para namespaces.

```
<detail>
      XML is not wellformed:
      The prefix "ipgapi" for element "ipgapi:IPGApiOrderRequest"
      is not bound.
</detail>
```

Explicação possível:
O namespace “ipgapi” não foi declarado. Para declarar um namespace, use o prefixo xmlns. Nesse caso, você deveria usar `xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi"` como atributo na tag de nível superior da mensagem da API (IPGApiOrderRequest ou IPGApiActionRequest).

```
<detail>
      XML is not wellformed:
      The prefix "xmln" for attribute "xmln:ns2" associated
      with an element type "ns3:IPGApiOrderRequest" is not bound.
</detail>
```

Explicação possível:
Para declarar um namespace próprio, somente o namespace xmlns predefinido é permitido. Nesse caso, o prefixo é escrito como xmln, não xmlns.

```
<detail>
      XML is not wellformed:
      Unable to create envelope from given source
      because the namespace was not recognized
</detail>
```

Explicação possível:
A mensagem pode ser interpretada como uma mensagem XML e a mensagem de SOAP anexada está correta, mas a mensagem da API no corpo de SOAP não tem namespaces ou os namespaces não estão declarados corretamente. Os namespaces corretos são descritos no xsd.

```
<detail>
      XML is not wellformed:
      The processing instruction target matching "[xX][mM][lL]"
      is not allowed.
</detail>
```

Explicação possível:
A mensagem completa deve ser uma mensagem XML correta para que a mensagem da API não contenha a declaração de xml <?xml … ?>.

```
<detail>
      Unexpected characters before XML declaration
</detail>
```

Explicação possível:
O XML deve iniciar com “<?xml”. Confirme isso para não enviar uma linha em branco ou outro caractere vazio em frente do xml.

```
<detail>
      XML is not a SOAP message:
      Unable to create envelope from given source
      because the root element is not named "Envelope"
</detail>
```

Explicação possível:
A mensagem parece ser uma mensagem XML correta, mas somente mensagens SOAP são aceitas. Essa mensagem deve ser anexada com uma mensagem de SOAP.

```
<detail>
      XML is not a valid SOAP message:
      Error with the determination of the type.
      Probably the envelope part is not correct.
</detail>
```

Explicação possível:
A tag do corpo de SOAP está ausente.

```
<detail>
      Source object passed to ''{0}'' has no contents.
</detail>
```

Explicação possível:
O corpo de SOAP está vazio. A mensagem da API anexa está ausente.

```
<detail>
      Included XML is not a valid IPG API message:
      unsupported top level {namespace}tag "irgendwas" in the soap body.
      Only one of [
      {http://ipg-online.com/ipgapi/schemas/ipgapi}IPGApiActionRequest,
      {http://ipg-online.com/ipgapi/schemas/ipgapi}IPGApiOrderRequest
      ] allowed.
</detail>
```

Explicação possível:
A primeira tag na mensagem da API anexa deve ser uma tag IPGApiActionRequest ou IPGApiOrderRequest e não a tag irgendwas. Neste caso, essa tag não tem namespace.

```
<detail>
      Included XML is not a valid IPG API message:
      unsupported top level {namespace}tag
      "{http://firstdata.de/ipgapi/schemas/ipgapi}IPGApiOrderRequest" in
      the soap body. Only one of [
      {http://ipg-online.com/ipgapi/schemas/ipgapi}IPGApiActionRequest,
      {http://ipg-online.com/ipgapi/schemas/ipgapi}IPGApiOrderRequest
      ] allowed.
</detail>
```

Explicação possível:
A tag de nível superior da mensagem da API não tem a tag permitida. Neste caso, o espaço do nome está errado.

```
<detail>
      cvc-pattern-valid:
      Value '1.234' is not facet-valid with respect to pattern
      '([1-9]([0-9]{0,12}))?[0-9](\.[0-9]{1,2})?' for type
      '#AnonType_ChargeTotalAmount'
      cvc-type.3.1.3:
      The value '1.234' of element 'ns3:ChargeTotal' is not valid.
</detail>
```

Explicação possível:
O valor de uma tag não corresponde à declaração no xsd. O valor tem três casas decimais, mas o xsd somente tem duas.

```
<detail>
      cvc-complex-type.2.4.a:
      Invalid content was found starting with element 'ns2:ExpYear'.
      One of '{"http://ipg-online.com/ipgapi/schemas/v1":ExpMonth}'
      is expected.
</detail>
```

Explicação possível:
As ocorrências das tags devem corresponder ao xsd. Recomendamos usar as tags na mesma sequência conforme declaradas no xsd. Neste caso, a tag esperada é ExpMonth, não ExpYear.

20.2 Resolução de problemas – Como processar exceções

```
<detail>
      <ipgapi:IPGApiOrderResponse
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
            <ipgapi:CommercialServiceProvider />
            <ipgapi:TransactionTime>1233656751183</ipgapi:TransactionTime>
            <ipgapi:ProcessorReferenceNumber />
            <ipgapi:ProcessorResponseMessage />
            <ipgapi:ErrorMessa
                  ge> SGS-C:
                  000003:
                  illegal comSicrediation of values for the
                  3DSecure: (VerificationResponse,
                  PayerAuthenticationResponse,
                  PayerAuthenticationCode) N Y null
            </ipgapi:ErrorMessage>
            <ipgapi:OrderId />
            <ipgapi:ApprovalCode />
            <ipgapi:AVSResponse />
            <ipgapi:TDate />
            <ipgapi:TransactionResult>FAILED</ipgapi:TransactionResult>
            <ipgapi:TerminalID />
            <ipgapi:ProcessorResponseCode />
            <ipgapi:ProcessorApprovalCode />
            <ipgapi:ProcessorReceiptNumber />
            <ipgapi:ProcessorTraceNumber />
      </ipgapi:IPGApiOrderRespon
                  se>
</detail>
```

Explicação:
A comSicrediação dos três valores VerificationResponse, PayerAuthenticationResponse e PayerAuthenticationCode para 3DSecure está errada. As comSicrediações permitidas são

Verification-Response	Payer-Authentication-Response	Payer-Authentication-Code	Código de resposta 3dsecure	Comentários
null	null	null	N/A	A transação será passada para o sistema de autorização sem informações do 3dsecure Sem MC ECI, Visa ECI = 7
N	null	null	7	Portador de cartão não inscrito Sem MC ECI, Visa ECI = 6
N	N	null	7	Portador de cartão não inscrito Sem MC ECI, Visa ECI = 6
U	null	null	5	Impossível autenticar (DS não acessível) Sem MC ECI, Visa ECI = 7
Y	A	null	4	Tentativa (o ACS não pode informar o resultado da autenticação) MC ECI = 1, Visa ECI = 6
Y	A	x	4	Tentativa (o ACS não pode informar o resultado da autenticação) MC ECI = 1, Visa ECI = 6
Y	U	null	6	Impossível autenticar (ACS não acessível) Sem MC ECI, Visa ECI = 7
Y	Y	null	2	Sucesso de autorização (sem CAAV / UCAF) MC ECI = 2, Visa ECI = 5
Y	Y	x	1	Sucesso de autorização MC ECI = 2, Visa ECI = 5

Y	N	null	3	Falha na autorização (Verificação de assinatura incorreta) - IPG recusa a transação ("N:-5101:3D Secure authentication failed") Sem MC ou Visa ECI

Outras comSicrediações não listadas acima serão recusadas pelo Sicredi e-commerce com um código 3D Secure de resposta = 8 e "N:-5100:Invalid 3D Secure values".

XID (criado pelo MPI antes de enviar a solicitação de verificação) precisa ser definido para transação VISA.

O código de autenticação do pagador x significa que o valor não é nulo.

```
<detail>
     <ipgapi:IPGApiOrderResponse
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
            <ipgapi:CommercialServiceProvider />
            <ipgapi:TransactionTime>1233656752933</ipgapi:TransactionTime>
            <ipgapi:ProcessorReferenceNumber />
            <ipgapi:ProcessorResponseMessage />
            <ipgapi:ErrorMessage>
                  SGS-005005: Duplicate transaction.
            </ipgapi:ErrorMessage>
            <ipgapi:OrderId>
                  IPGAPI-REQUEST-29351d8e-2634-4725-9d93-91b83704e00d
            </ipgapi:OrderId>
            <ipgapi:ApprovalCode />
            <ipgapi:AVSResponse />
            <ipgapi:TDate />
            <ipgapi:TransactionResult>FRAUD</ipgapi:TransactionResult>
            <ipgapi:TerminalID />
            <ipgapi:ProcessorResponseCode />
            <ipgapi:ProcessorApprovalCode />
            <ipgapi:ProcessorReceiptNumber />
            <ipgapi:ProcessorTraceNumber />
      </ipgapi:IPGApiOrderRespon
                  se>
</detail>
```

Explicação:
Depois da primeira transação, as demais com os mesmos dados são bloqueadas por um período configurável. Consulte o Guia de usuário do Terminal Virtual para obter detalhes sobre configurações de fraude.

```
<detail>
     <ipgapi:IPGApiOrderResponse
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
            <ipgapi:CommercialServiceProvider />
            <ipgapi:TransactionTime>1233656752308</ipgapi:TransactionTime>
            <ipgapi:ProcessorReferenceNumber />
            <ipgapi:ProcessorResponseMessage />
            <ipgapi:ErrorMessage>
                  SGS-005009:
                  The currency is not allowed for this terminal.
            </ipgapi:ErrorMessage>
            <ipgapi:OrderId>
                  IPGAPI-REQUEST-a58f6631-eb71-49c8-bbca-23fff53252fc
            </ipgapi:OrderId>
            <ipgapi:ApprovalCode />
            <ipgapi:AVSResponse />
            <ipgapi:TDate />
            <ipgapi:TransactionResult>FAILED</ipgapi:TransactionResult>
            <ipgapi:TerminalID />
            <ipgapi:ProcessorResponseCode />
            <ipgapi:ProcessorApprovalCode />
            <ipgapi:ProcessorReceiptNumber />
            <ipgapi:ProcessorTraceNumber />
      </ipgapi:IPGApiOrderResponse>
</detail>
```

Explicação:
Este é um exemplo com dólares americanos, que não é a moeda permitida para esta loja.

```
<detail>
      <ipgapi:IPGApiOrderResponse
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
            <ipgapi:CommercialServiceProvider />
            <ipgapi:TransactionTime>1234346305732</ipgapi:TransactionTime>
            <ipgapi:ProcessorReferenceNumber />
            <ipgapi:ProcessorResponseMessage />
            <ipgapi:ErrorMessage>
                  SGS-032000: Unknown processor error occured.
            </ipgapi:ErrorMessage>
            <ipgapi:OrderId>
                  IPGAPI-REQUEST-b3223ee5-156b-4d22-bc3f-910709d59202
            </ipgapi:OrderId>
            <ipgapi:ApprovalCode />
            <ipgapi:AVSResponse />
            <ipgapi:TDate>1234346284</ipgapi:TDate>
            <ipgapi:TransactionResult>DECLINED</ipgapi:TransactionResult>
            <ipgapi:TerminalID />
            <ipgapi:ProcessorResponseCode />
            <ipgapi:ProcessorApprovalCode />
            <ipgapi:ProcessorReceiptNumber />
            <ipgapi:ProcessorTraceNumber />
      </ipgapi:IPGApiOrderResponse>
</detail>
```

Explicação:
Se suas transações são executadas normalmente, uma explicação possível é que o número dos IDs de Terminal atribuídos à sua loja não são suficientes para o volume de transações. Entre em contato com a equipe de Vendas para solicitar mais Números de terminal para equilibrar a carga.

20.3 Resolução de problemas – Mensagens de erro de login ao usar o cURL

```
* About to connect() to test.ipg-online.com port 443 (#0)
*   Trying 217.73.32.55... connected
* Connected to test.ipg-online.com (217.73.32.55) port 443 (#0)
* unable to set private key file: 'C:\API\config\WS120666668._.1.key'
type PEM
* Closing connection #0
curl: (58) unable to set private key file:
'C:\API\config\WS120666668._.1.key' type
PEM
```

Explicação:
A keystore e a senha não coincidem. Verifique se você usou a keystore e a senha correta. Verifique se você usou o arquivo **WS**<storeId>._.1.pem. Se você incluir .cer ao nome do arquivo, você pode abrir o certificado com um duplo clique. O certificado deve ser exposto para sua loja. Remove a extensão .cer depois da verificação.

```
* SSL certificate problem, verify that the CA cert is OK. Details:
error:14090086:SSL routines:SSL3_GET_SERVER_CERTIFICATE:certificate
verify failed
* Closing connection #0
curl: (60) SSL certificate problem, verify that the CA cert is OK.
Details: error:14090086:SSL
routines:SSL3_GET_SERVER_CERTIFICATE:certificate verify failed
More details here: http://curl.haxx.se/docs/sslcerts.html

curl performs SSL certificate verification by default, using a "bundle"
of Certificate Authority (CA) public keys (CA certs). The default bundle
is named curl-ca-bundle.crt; you can specify an alternate file using the
-- cacert option.
If this HTTPS server uses a certificate signed by a CA represented in
the bundle, the certificate verification probably failed due to a
problem with the certificate (it might be expired, or the name might not
match the domain name in the URL).
If you'd like to turn off curl's verification of the certificate, use the
- k (or --insecure) option
```

Explicação:
O certificado de truststore está incorreto. Verifique o truststore: inclua .cer ao nome do arquivo geotrust.pem e abra o certificado com um duplo clique. Você deve ver o emissor Equifax.
Altere o nome geotrust.pem.cer depois do teste de volta para geotrust.pem.

```
<html>
      <head>
            <title>Apache Tomcat/5.5.20 - Error report</title>
            <style>
                  <!--
H1 {font-family:Tahoma,Arial,sans-
serif;color:white;background- color:#525D76;font-size:22px;}
H2 {font-family:Tahoma,Arial,sans-
serif;color:white;background- color:#525D76;font-size:16px;}
H3 {font-family:Tahoma,Arial,sans-
serif;color:white;background- color:#525D76;font-size:14px;}
BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-
color:white;}
B {font-family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;}
P {font-family:Tahoma,Arial,sans-serif;background:white;color:black;font-
size:12px;}
A {color : black;}
A.name {color : black;}
HR {color : #525D76;}
                  -->
            </style>
      </head>
      <body>
            <h1>HTTP Status 401 - </h1>
            <HR size="1" noshade="noshade">
            <p><b>type</b> Status report</p><p><b>message</b>
                  <u></u></p><p><b>description</b>
                  <u>This request requires HTTP authentication ().</u></p>
            <HR size="1" noshade="noshade">
            <h3>Apache Tomcat/5.5.20</h3>
      </body>
</html>
```

Explicação:
Seus certificados estão em ordem e foram aceitos, mas sua senha ou usuário está errado.

20.4 Resolução de problemas – Mensagens de erro de login ao usar o Java client

```
java.io.IOException: Keystore was tampered with, or password was
incorrect
```

Explicação:
Sua senha de keystore não coincide com a keystore ou a senha de truststore com a trustore. Você pode verificar a senha com a keytool que é um componente do JDK. Você pode encontrá-la no diretório de Sicredi do JDK. Para testar a chamada de senha c:\Programme\Java\jdk1.6.0_07\Sicredi\keytool.exe -list -v -keystore <your keystore or truststore> -storepass <your keystore or truststore password>

```
javax.net.ssl.SSLHandshakeException:
sun.security.validator.ValidatorException: No trusted certificate
found
```

Explicação:
Sua truststore está incorreta. Você pode inspecionar sua truststore com a keytool, um componente JDK:
`c:\Programme\Java\jdk1.6.0_07\Sicredi\keytool.exe -list -v -keystore` <your truststore> `-storepass` <your truststore password>
e você verá o emissor Equifax
OU=Equifax Secure Certificate Authority, O=Equifax, C=US no resultado. Verifique os valores de MD5 e SHA1 também.

```
<html>
      <head>
            <title>Apache Tomcat/5.5.20 - Error report</title>
            <style><!--H1 {font-family:Tahoma,Arial,sans-
serif;color:white;background-color:#525D76;font-size:22px;} H2 {font-
family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;font- size:16px;} H3 {font-family:Tahoma,Arial,sans-
serif;color:white;background- color:#525D76;font-size:14px;} BODY {font-
family:Tahoma,Arial,sans- serif;color:black;background-color:white;} B
{font- family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;} P
{font-family:Tahoma,Arial,sans-serif;background:white;color:black;font-
size:12px;}A {color : black;}A.name {color : black;}HR {color :
#525D76;}--></style>
      </head>
      <body>
            <h1>HTTP Status 401 -</h1>
            <HR size="1" noshade="noshade">
            <p>
                  <b>type</b>
                  Status report
            </p>
            <p>
                  <b>message</b>
                  <u></u>
            </p>
            <p>
                  <b>description</b>
                  <u>This request requires HTTP authentication ().</u>
            </p>
             <HR size="1" noshade="noshade">
             <h3>Apache Tomcat/5.5.20</h3>
      </body>
</html>
```

Explicação:
Seu ID do Usuário ou senha está incorreta.