



GPS Check Processing SOAP Web Service

Version 2.0

Guardian Payment Systems

Guardian Payment Systems SOAP Web Service for processing batches of both Automated Clearing House (ACH) and Remote Deposit Capture (RDC) Check transactions.

Contents

Guardian Payment Systems Check Processing	1
Client boarding and settlement banking	1
Technical Details	1
ACH Batch Processing	2
Submitting a Check Batch to the Web Service: SubmitCheckBatch(...)	2
Check Batch Header Object	2
Check Batch Request Object	2
Check Batch Object	2
Check Object	2
Check Batch Response Object.....	3
ACH Batch Status during Processing: GetCheckBatchStatusSummary (...).....	3
Check Batch Header Object	4
Check Batch Status Summary Request Object.....	4
Check Batch Status Summary Response Object	4
Check Status Summary Object.....	4
ACH Batch Results: GetCheckBatchResults(...)	5
Check Batch Header Object	5
Check Batch Result Request Object	5
Check Batch Result Response Object.....	5
Check Results Object.....	5
RDC Batch Processing	7
Submitting a Check Batch to the Web Service: SubmitCheckBatch(...)	7

GPS Check Processing SOAP Web Service

Check Object	7
Appendices: Example Code.....	8
Appendix A - Create the ACH Check Batch	8
Appendix B - Create the Web Service Objects.....	8
Appendix C - Submit the Batch to the Web Service.....	9
Appendix D - Monitor the Batch until Processing is Complete.....	9
Appendix E – Processing a Batch with 1 ACH Transaction.....	10

Guardian Payment Systems Check Processing

The Guardian Payment Systems Check Processing system provides an SOAP Web Service for customers to be able to process Check transactions; these transactions can be RDC transactions, where a paper check is scanned by a check imager resulting in the electronic version of the check replacing/superseding the paper check. These transactions can be tradition ACH transactions where funds are transferred electronically from the bank account.

This service provides the following methods:

- SubmitCheckBatch: submit a check batch to the SOAP Web Service for processing
- GetCheckBatchResults: get the results or current status of a check batch submitted to the SOAP Web Service for processing

This document outlines the process that a customer needs to do to use these services, but these services can also be consumed via XML/POX Services.

Client boarding and settlement banking

Merchants must be boarded as a client with Guardian Payment Systems prior to initiating check processing through the check processing interface. The boarding process includes being underwritten by a bank settlement institution. Guardian supports the following settlement institutions:

- Check Assist
- Frost Bank
- Other settlement institutions can be provided for you by Guardian's Professional Services Team.

Technical Details

This service is coded in .Net 3.5 and is exposed as an XML Web Service. The WSDL is:

<https://testsvcs.guardianpayments.com/Processing/CheckBatchService.svc?wsdl>

ACH Batch Processing

Processing batches of ACH transactions follows this basic outline:

1. Submit your batch to the Guardian Web Service, after
 - a. Creating a batch object containing your check transactions, and
 - b. Creating the objects to interact with the Guardian Web Service , and
2. Monitor until the Guardian Web Service indicates the batch processing is complete.

Submitting a Check Batch to the Web Service: SubmitCheckBatch(...)

Submitting a check batch through the SOAP Web service follows this pseudo-syntax:

```

CheckBatchResponse = CheckBatchWebService.SubmitCheckBatch
(
    CheckMessageHeader,
    CheckBatchRequest
)
    
```

Check Batch Header Object

The header object contains information to identify you as a merchant within Guardian’s Payment Director Payment gateway. These merchant specific values are provided by Guardian when merchant underwriting is complete. An example creating and populating the header can be found in Appendices B and E.

Check Batch Request Object

The CheckBatchRequest object will contain the specific information on the batch and the check transactions that are being submitted for processing. The CheckBatchRequest object is defined as:

Field Name	Type	Description
CheckBatch	CheckBatch	The transactions and data being submitted for processing.

Check Batch Object

The Guardian CheckBatch object is defined as follows:

Field Name	Type	Description
ClientBatchIdentifier	string(50)	Required. Your system’s reference number to identify this check batch.
Checks	Check[]	Required. An array of Check objects, each one referring to a single settlement request. NOTE: In Visual Studio, this can be a List if defined as such in the “Dictionary collection type” of the Service Reference Configuration.
TransactionCount	int	Required. Count of all transactions in the Checks array
TotalAmount	decimal	Required. Sum of the amount of each transaction in the Checks array

Check Object

The Check object is defined as follows for ACH transactions:

Field Name	Type	Description
ClientTransaction Identifier	string(50)	Required. Your system’s reference number to identify this transaction.

GPS Check Processing SOAP Web Service

Field Name	Type	Description
TransactionType	enum	Required. CCD, TEL, WEB, PPD
AccountHolderName	String(100)	Required. Account holder name
RoutingNumber	String(9)	Required. 9 digit ABA Routing Number
AccountNumber	String(55)	Required. Bank Account Number
VirtualCheckAccountId entifier	String	Optional. Guardian ID supplied by the Virtual Vault Web Service
Amount	Decimal	Required. Transaction amount
MICR	String(150)	Ignored. Magnetic read from a check scanner of the MICR value on the paper check
CheckNumber	String(15)	Ignored. Check number on the paper check
FrontImage	Image	Ignored. TIFF image captured by a check scanner of the front of the paper check
BackImage	Image	Ignored. TIFF image captured by a check scanner of the back of the paper check
AccountType	String(20)	Required. Either "Checking" or "Savings"
EntryType	String(10)	Optional. Either "Debit" or "Credit"
ClientData1	String(50)	Optional. A value from your system that will show up in Guardian Reporting. Examples include Customer Id, Donor Id, Order Description, etc.
ClientData2	String(50)	Optional. A value from your system that will show up in Guardian Reporting.
ClientData3	String(50)	Optional. A value from your system that will show up in Guardian Reporting.

Check Batch Response Object

Field Name	Type	Description
ClientBatchIdentifier	string(50)	Echo of your system's batch Id that was supplied in the Check Batch object.
BatchIdentifier	String(50)	Guardian's id for this batch.

ACH Batch Status during Processing: [GetCheckBatchStatusSummary \(...\)](#)

While the batch is processing, a call to the `GetCheckBatchStatusSummary` method can retrieve intermediate results. This is entirely optional, and if you have large batches, you may wish to invoke a timer to inquire on the batch status to run every minute. This gives you counts of the ongoing progress of this batch. When the batch is complete, the `BatchStatus` that is returned from this call will be "Complete". Additionally you receive an array of all current transaction status and their counts. So, if all transactions show Invalid, you will know that you have a pretty serious error in your original transaction data.

The Web Service call to get the batch results follows this pseudo-syntax:

```
CheckBatchStatusSummaryResponse = CheckBatchServiceClient.GetCheckBatchStatusSummary
(
    CheckMessageHeader,
    CheckBatchStatusSummaryRequest
)
```

Check Batch Header Object

As was the case when submitting the batch, the header object for getting batch status contains information to identify you as a merchant within Guardian’s Payment Director Payment gateway, and examples for creating and populating the header can be found in the Appendices.

Check Batch Status Summary Request Object

The CheckBatchStatusSummaryRequest object will contain the identifying information on the batch and is defined as:

Field Name	Type	Description
ClientBatchIdentifier	string(50)	Optional. Echo of your system’s batch Id that was supplied in the Check Batch object.
BatchIdentifier	String(50)	Optional. Guardian’s id for this batch.

Either the ClientBatchIdentifier or the BatchIdentifier must be provided.

Check Batch Status Summary Response Object

The CheckBatchStatusSummaryResponse object is defined as follows:

Field Name	Type	Description
ClientBatchIdentifier	String	Your system’s reference number to identify this check batch.
BatchIdentifier	String	Guardian’s Batch Identifier
BatchStatus	String	Batch’s current status: <ul style="list-style-type: none"> • Received: batch has been received, waiting for validation • Validating: batch is being validated • Validated: batch has been validated, is waiting for processing • Processing: batch is being processed • Complete: batch has been processed and will be placed in the next appropriate file for transmission to the bank.
CheckStatusSummaryList	CheckStatusSummary[]	Array of CheckStatusSummary objects

Check Status Summary Object

The CheckStatusSummary object contains the status result groupings for every transaction in the batch and is defined as:

Field Name	Type	Description
Status	String	Transaction status: <ul style="list-style-type: none"> • Received: the transaction has been received, is waiting of processing • Invalid: Guardian marked the transaction invalid (e.g. invalid routing number) • Processing: transaction is processing • Failed: transaction has completed resulting in an error • Complete: transaction processing has finished normally
Count	Int	Count of transactions with the given status.

ACH Batch Results: GetCheckBatchResults(...)

Once the batch processing is complete at Guardian Payment System’s Payment Director payment gateway, you can download the results of the specific batch. The amount of time required to process the batch varies based on batch size, so if you attempt to retrieve the results too soon, you will receive an exception stating that the batch is currently being processed.

The Web Service call to get the batch results follows this pseudo-syntax:

```

CheckBatchResultResponse = CheckBatchServiceClient.GetCheckBatchResults
(
    CheckMessageHeader,
    CheckBatchResultRequest
)
    
```

Check Batch Header Object

As was the case when submitting the batch, the header object for getting batch status contains information to identify you as a merchant within Guardian’s Payment Director Payment gateway, and examples for creating and populating the header can be found in the Appendices.

Check Batch Result Request Object

The CheckBatchResultRequest object will contain the identifying information on the batch and is defined as:

Field Name	Type	Description
ClientBatchIdentifier	string(50)	Optional. Echo of your system’s batch Id that was supplied in the Check Batch object.
BatchIdentifier	String(50)	Optional. Guardian’s id for this batch.

Either the ClientBatchIdentifier or the BatchIdentifier must be provided.

Check Batch Result Response Object

The CheckBatchResultResponse object is defined as follows:

Field Name	Type	Description
ClientBatchIdentifier	String	Your system’s reference number to identify this check batch.
BatchIdentifier	String	Guardian’s Batch Identifier
CheckResults	CheckResult[]	Array of CheckResult objects – 1 for each transaction submitted

Check Results Object

The CheckResult object contains the result of every transaction in the batch and is defined as:

Field Name	Type	Description
ClientTransactionIdentifier	String	Your system’s reference number to identify this check transaction.
TransactionIdentifier	String	Guardian’s transaction Identifier
ResponseCode	String	Transaction result
ResponseMessage	String	
TransactionDate	DateTime	
Status	String	Transaction status:

GPS Check Processing SOAP Web Service

Field Name	Type	Description
		<ul style="list-style-type: none">• Received: transaction has been received, waiting for processing• Invalid: Guardian marked the transaction invalid (e.g. invalid routing number)• Processing: transaction is processing• Failed: transaction has completed resulting in an error• Complete: transaction processing has finished normally• Returned: the bank has returned this transaction• Canceled: At your request, Guardian canceled the batch containing this transaction• Hold: At your request, Guardian has put the batch containing this transaction on hold

RDC Batch Processing

Processing of your RDC transactions follows the ACH processing outlined above, except where noted in this section.

Submitting a Check Batch to the Web Service: SubmitCheckBatch(...)

Submitting a batch or RDC transactions through the SOAP Web service follows the description of ACH transaction processing outlined above with the exception of the Check object.

Check Object

The Check object is defined as follows for ACH transactions:

Field Name	Type	Description
ClientTransaction Identifier	string(50)	Required. Your system's reference number to identify this transaction.
TransactionType	enum	Required. Check21
AccountHolderName	String(100)	Optional. Account holder name
RoutingNumber	String(9)	Ignored. 9 digit ABA Routing Number
AccountNumber	String(55)	Ignored. Bank Account Number
VirtualCheckAccountIdentifier	String	Ignored.
Amount	Decimal	Required. Transaction amount
MICR	String(150)	Required. Magnetic read from a check scanner of the MICR value on the paper check
CheckNumber	String(15)	Optional. Check number on the paper check
FrontImage	Image	Required. TIFF image captured by a check scanner of the front of the paper check
BackImage	Image	Required. TIFF image captured by a check scanner of the back of the paper check
AccountType	String(20)	Ignored.
EntryType	String(10)	Ignored.
ClientData1	String(50)	Optional. A value from your system that will show up in Guardian Reporting. Examples include Customer Id, Donor Id, Order Description, etc.
ClientData2	String(50)	Optional. A value from your system that will show up in Guardian Reporting.
ClientData3	String(50)	Optional. A value from your system that will show up in Guardian Reporting.

Appendices: Example Code

The section shows example code for the 4 steps outlined above, as well as a complete listing of this example code.

Appendix A - Create the ACH Check Batch

The code to create an ACH check batch and populate it with a single transaction follows:

```
// ***** STEP 1: Create the check batch
// Create the CheckBatch to hold our 1 transaction;
// place your system's batch Id in the ClientBatchIdentifier
GPSCheckBatchSoapWebService.CheckBatch CheckBatch = new GPSCheckBatchSoapWebService.CheckBatch();
CheckBatch.ClientBatchIdentifier =
createUniqueClientAchBatchId(achWebPayment.MerchantTransactionId.ToString());

// Place the single ACH transaction into a Check object
CheckBatch.Checks = new List<GPSCheckBatchSoapWebService.Check>();
GPSCheckBatchSoapWebService.Check check = new GPSCheckBatchSoapWebService.Check();
// Place your transaction Id in the ClientTransactionIdentifier
check.ClientTransactionIdentifier = achWebPayment.MerchantTransactionId.ToString();
check.AccountHolderName = achWebPayment.AccountHolderName;
check.Amount = achWebPayment.Amount;
check.AccountType = (GPSCheckBatchSoapWebService.AccountType)achWebPayment.AccountType;

// This example demonstrates sending ACH Debit transactions
check.TransactionType = GPSCheckBatchSoapWebService.TransactionType.WEB;
check.EntryType = GPSCheckBatchSoapWebService.EntryType.Debit;

// If this is this a Virtual Account Vault transaction, meaning the bank account information
// has already been stored in the Guardian Vault, then use the Vault Token instead of the
// Bank information. Refer to the GPS Virtual Account Vault SOAP Web Service documentation
// for more information.
if (achWebPayment.VirtualAccountId != Guid.Empty)
    check.VirtualCheckAccountIdentifier = achWebPayment.VirtualAccountId.ToString();
else
{
    // Test Routing Numbers: 222371863 307075259 052000113
    check.RoutingNumber = achWebPayment.RoutingNumber;
    check.AccountNumber = achWebPayment.AccountNumber;
}

// Place the single transaction in the check batch created above.
CheckBatch.Checks.Add(check);

// Set the Batch Totals
CheckBatch.TotalAmount = check.Amount;
CheckBatch.TransactionCount = 1;
```

Appendix B - Create the Web Service Objects

```
// ***** STEP 2: Create the objects to interact with the Web Service
// Create the Service Client object, the Check Message Header Object,
// and the Check Message Request object
GPSCheckBatchSoapWebService.CheckBatchServiceClient client =
    new GPSCheckBatchSoapWebService.CheckBatchServiceClient();
GPSCheckBatchSoapWebService.CheckMessageHeader Header =
    new GPSCheckBatchSoapWebService.CheckMessageHeader();
// These values are Provided by Guardian when the Merchant Boarding Process is completed
Header.ActivationCode = achWebPayment.GuardianActivation.ToString();
Header.EndpointToken = achWebPayment.GuardianEndpoint.ToString();
GPSCheckBatchSoapWebService.CheckBatchRequest Request = new
GPSCheckBatchSoapWebService.CheckBatchRequest();
Request.CheckBatch = CheckBatch;
```

Appendix C - Submit the Batch to the Web Service

```
// ***** STEP 3: Submit the batch
// Submit the batch to the Web Service; store the response object to monitor the batch
GPSCheckBatchSoapWebService.CheckBatchResponse Response =
    client.SubmitCheckBatch(Header, Request);
resultBatchId = Response.BatchIdentifier;
```

Appendix D - Monitor the Batch until Processing is Complete

There are many ways you can construct your application to get the batch results when processing is completed, but polling until it is complete is the most common method. Devise the method best suited for your application:

- Polling for results by calling GetCheckBatchResults,
- Getting interim processing status by calling GetCheckBatchStatusSummary, and then calling GetCheckBatchResults when the batch's status summary is "Complete", or
- Waiting an appropriate amount of time and calling GetCheckBatchResults.

The call to get the batch results follows:

```
// Call the Batch Web Service to fetch the results of the transmission
GPSCheckBatchSoapWebService.CheckBatchResultRequest resultRequest =
    new GPSCheckBatchSoapWebService.CheckBatchResultRequest();
resultRequest.BatchIdentifier = resultBatchId;
GPSCheckBatchSoapWebService.CheckBatchResultResponse response =
    client.GetCheckBatchResults(Header, resultRequest);
```

Appendix E – Processing a Batch with 1 ACH Transaction

```
/// <summary>
/// This example code demonstrates how to process a single ACH payment through the
/// GPS Check Processing SOAP Web Service. The 4 steps in the process are:
/// Step 1: Create the Check Batch
/// Step 2: Create the objects to interact with the Web Service
/// Step 3: Submit the batch
/// Step 4: Monitor until the batch completes processing
/// In this example the "Client" is this method representing your application, which is the
/// client from the perspective of the Guardian Web Service.
/// </summary>
/// <param name="achWebPayment">
///     string RoutingNumber
///     string AccountNumber
///     string AccountHolderName
///     string CheckNumber
///     AccountType: enum including 'Checking' and 'Savings'
/// </param>
/// <returns>TODO: determine return value, if any, based on your client application</returns>
private string doAchBatch(AchWebPaymentItem achWebPayment)
{
    string resultBatchId = string.Empty;
    string resultTransactionId = string.Empty;

    // ***** STEP 1: Create the check batch
    // Create the CheckBatch to hold our 1 transaction;
    // place your system's batch Id in the ClientBatchIdentifier
    GPSCheckBatchSoapWebService.CheckBatch CheckBatch =
        new GPSCheckBatchSoapWebService.CheckBatch();
    CheckBatch.ClientBatchIdentifier =
        createUniqueClientAchBatchId(achWebPayment.MerchantTransactionId.ToString());

    // Place the single ACH transaction into a Check object
    CheckBatch.Checks = new List<GPSCheckBatchSoapWebService.Check>();
    GPSCheckBatchSoapWebService.Check check = new GPSCheckBatchSoapWebService.Check();
    // Place your transaction Id in the ClientTransactionIdentifier
    check.ClientTransactionIdentifier = achWebPayment.MerchantTransactionId.ToString();
    check.AccountHolderName = achWebPayment.AccountHolderName;
    check.Amount = achWebPayment.Amount;
    check.AccountType = (GPSCheckBatchSoapWebService.AccountType)achWebPayment.AccountType;

    // This example demonstrates sending ACH Debit transactions
    check.TransactionType = GPSCheckBatchSoapWebService.TransactionType.WEB;
    check.EntryType = GPSCheckBatchSoapWebService.EntryType.Debit;

    // If this is this a Virtual Account Vault transaction, meaning the bank account information
    // has already been stored in the Guardian Vault, then use the Vault Token instead of the
    // Bank information. Refer to the GPS Virtual Account Vault SOAP Web Service documentation
    // for more information.
    if (achWebPayment.VirtualAccountId != Guid.Empty)
        check.VirtualCheckAccountId = achWebPayment.VirtualAccountId.ToString();
    else
    {
        // Test Routing Numbers: 222371863 307075259 052000113
        check.RoutingNumber = achWebPayment.RoutingNumber;
        check.AccountNumber = achWebPayment.AccountNumber;
    }

    // Place the single transaction in the check batch created above.
    CheckBatch.Checks.Add(check);

    // Set the Batch Totals
    CheckBatch.TotalAmount = check.Amount;
    CheckBatch.TransactionCount = 1;

    // ***** STEP 2: Create the objects to interact with the Web Service
    // Create the Service Client object, the Check Message Header Object,
    // and the Check Message Request object
    GPSCheckBatchSoapWebService.CheckBatchServiceClient client =
```

GPS Check Processing SOAP Web Service

```
        new GPSCheckBatchSoapWebService.CheckBatchServiceClient();
GPSCheckBatchSoapWebService.CheckMessageHeader Header =
    new GPSCheckBatchSoapWebService.CheckMessageHeader();
// These values are Provided by Guardian when the Merchant Boarding Process is completed
Header.ActivationCode = achWebPayment.GuardianActivation.ToString();
Header.EndpointToken = achWebPayment.GuardianEndpoint.ToString();
GPSCheckBatchSoapWebService.CheckBatchRequest Request =
    new GPSCheckBatchSoapWebService.CheckBatchRequest();
Request.CheckBatch = CheckBatch;

// ***** STEP 3: Submit the batch
// Submit the batch to the Web Service; store the response object to monitor the batch
// as it is processed
GPSCheckBatchSoapWebService.CheckBatchResponse Response = client.SubmitCheckBatch(Header,
Request);
resultBatchId = Response.BatchIdentifier;

// ***** STEP 4: Monitor until the batch completes processing
// This example merely sleeps for 1 second then checks for results. Your client application
// will need a more robust method of polling until the response signals the batch has
completed.
System.Threading.Thread.Sleep(1000);

// Call the Batch Web Service to fetch the results of the transmission
// Alternatively, you can call GetCheckBatchStatusSummary to see intermidate results.
GPSCheckBatchSoapWebService.CheckBatchResultRequest resultRequest =
    new GPSCheckBatchSoapWebService.CheckBatchResultRequest();
resultRequest.BatchIdentifier = resultBatchId;
GPSCheckBatchSoapWebService.CheckBatchResultResponse response =
    client.GetCheckBatchResults(Header, resultRequest);

// Display the results of the results check
if (response != null)
{
    string clientBid = response.ClientBatchIdentifier;
    if (response.CheckResults.Count > 0)
    { // loop through each transaction, which in this case is only a single one
        string cTransID;
        string respCode = string.Empty;
        string respMsg = string.Empty;
        foreach (GPSCheckBatchSoapWebService.CheckResult result in response.CheckResults)
        {
            cTransID = result.ClientTransactionIdentifier;
            resultTransactionId = result.TransactionIdentifier;
            respCode = result.ResponseCode;
            respMsg = result.ResponseMessage;
        }
        if (respCode != "0")
        { // This example code just throws an exception if the batch processing failed.
          // Your client application will need to handle this appropriately.
            updatePaymentLog(achWebPayment.MerchantTransactionId,
                string.Format("Error {0} [{1}]", respCode, respMsg));
            throw new Exception(respMsg);
        }
    }
    else
    {
        //msg.Append("No Results returned for your batch!");
    }
}

return resultTransactionId;
}
```