PagOnline Imprese

Specifiche tecniche

Ver.12 del 01/12/2020

Banking that matters. | UniCredit

Prefazione

Scopo del documento

Il presente documento si propone di illustrare l'utilizzo dei Web Services o delle corrispondenti API, per l'integrazione con sistema di pagamento elettronico **PagOnline Imprese**.

A chi è rivolto

Il documento è rivolto ai programmatori che sviluppano applicazioni Web destinate all'integrazione con sistemi di pagamento elettronico.

Versione	Data	Note
01	15/05/2012	Prima versione
02	04/09/2012	
03	11/10/2012	
04	28/05/2013	
05	16/09/2013	
06	16/06/2014	Inseriti riferimenti MyBank
07	07/02/2017	Inseriti nuovi campi opzionali MyBank, sistemati esempi chiamate
08	02/11/2017	Inserita descrizione Selector
09	22/10/2018	Integrata descrizione Selector
10	01/02/2021	Inseriti riferimenti AccountToPay
10	01/02/2021	Inseriti riferimenti AccountToPay
11	10/05/2021	Inseriti codici di ritorno IGFS_20002, IGFS_20092; inseriti riferimenti al nuovo tipo di pagamento Bancomat Pay
12	01/12/2021	Inserite informazioni su tempistiche di rimborso per nuovi strumenti di pagamento

SOMMARIO

Introduzione

Nel presente documento vengono illustrate le specifiche tecniche dei Web Services e delle API, necessari per l'integrazione nel proprio portale web di eCommerce, con il sistema di pagamento elettronico **PagOnline Imprese** (Payment Gateway).

Gli strumenti di pagamento oggi disponibili sono:

- Carte di Credito (che include tutti i Circuiti abilitati Maestro, MasterCard, Visa Electron, Visa, VPAY e anche American Express e Diners se convenzionati);
- MyBank
- AccountToPay
- Bancomat Pay
- Alipay
- Apple Pay

Relativamente alle carte di credito tale opzione è integrata al prodotto base PagOnline opzione PA-DSS.

Gli unici dati sensibili memorizzati dai programmi sono il PAN carta e la Data scadenza della carta. Non sono invece rilevanti per i programmi in oggetto di certificazione i Nomi dei titolari della carta, il Codice di servizio, né sono memorizzati i dati completi della striscia magnetica e i CAV2/CID/CVC2/CVV2, come non lo sono il PIN e il PIN Block.

Per ogni documentazione riguardo gli standard PA-DSS si faccia riferimento al sito:
https://www.pcisecuritystandards.org/

Relativamente allo strumento di pagamento MyBank si faccia riferimento al sito:
www.mybank.eu

1 Descrizione del contesto di integrazione

La piattaforma di pagamento elettronico **PagOnline Imprese** è strutturata per offrire all'Esercente una modalità di integrazione basata su Open API denominata Selector, che semplifica l'integrazione con il Payment Gateway e permette l'integrazione dei nuovi strumenti di pagamento che la Banca metterà a disposizione tempo per tempo, senza la necessità di nuove implementazioni sul sito/App del Esercente.

La modalità Selector consente all'Esercente, titolare del servizio PagOnline Imprese eCommerce, di avere un unico punto di integrazione con il Payment Gateway di UniCredit. Gli Esercenti saranno pertanto abilitati automaticamente all'accettazione degli strumenti di pagamento convenzionati:

- Carte di Credito (che include tutti i Circuiti abilitati Maestro, MasterCard, Visa Electron, Visa, VPAY e anche American Express e Diners se convenzionati);
- MyBank;
- AccountToPay
- Bancomat Pay
- Alipay
- Apple Pay
- tutti gli altri strumenti di pagamento tempo per tempo integrati dalla Banca.

Il Selector rappresenta pertanto un aggregatore degli strumenti di pagamento, volto a semplificare l'integrazione da parte dell'Esercente del sito/App eCommerce con la piattaforma di pagamento.

Dal punto di vista dell'esperienza dell'acquirente (cliente), quest'ultimo, al momento del check out sul sito/App eCommerce dell'Esercente, dopo aver premuto il tasto "Paga" o altro di medesimo significato, potrà accedere all'ambiente securitizzato del Payment Gateway UniCredit dove visualizzerà l'elenco di tutti gli strumenti di pagamento che potrà selezionare per perfezionare l'acquisto.

In alternativa alla pagina del Selector, l'Esercente, in fase di integrazione con il Payment Gateway, avrà sempre la possibilità di scegliere di sviluppare le chiamate ai singoli strumenti di pagamento utilizzando l'apposito TID diversificato per ciascuno, per i quali dovrà andare a esporre sul proprio sito/App, in luogo del tasto "Paga", i loghi corrispondenti.

E' altresì possibile adottare una soluzione di tipo intermedio che fa coesistere il tasto "Paga" (chiamata al Selector che mostra la pagina degli strumenti di pagamento al cliente) e il tasto del singolo strumento di pagamento con relativo logo (ad es.: Bancomat Pay).

E' altresì possibile chiedere al servizio di BackOffice di PagOnline Imprese di rimuovere dalla pagina degli Strumenti di Pagamento del Selector un singolo strumento di pagamento (ad es.: Bancomat Pay) per il quale l'Esercente avrà scelto di sviluppare la chiamata direttamente al Payment Gateway.

Il **principale beneficio derivante dall'utilizzo del Selector**, che ricordiamo prevede l'accesso del cliente finale ad una pagina degli Strumenti di Pagamento attivabile dal sito/App dell'Esercente da unico tasto "Paga" (o altro tasto di medesimo significato), è che **svincola l'Esercente da nuove implementazioni derivanti tempo per tempo dall'introduzione di nuovi strumenti di pagamento** che la Banca metterà a disposizione degli Esercenti convenzionati e dotati della piattaforma PagOnline Imprese.
Le implementazioni per l'introduzione di nuovi strumenti di pagamento restano a carico di UniCredit sulla piattaforma PagoOnline Imprese.

ATTENZIONE: diverse software house hanno dei plug-in predefiniti che prevedono di default solo l'esposizione del pagamento con Carte (e i relativi loghi) e non considerano ad es. MyBank o AccountToPay o Bancomat Pay anch'essi disponibili nel Selector.

Esempio: modalità Selector con attive le soluzioni di incasso Carte, e MyBank/AccountToPay/Bancomat Pay
Nel caso di sottoscrizione sia dei Servizi di Acquiring che della soluzione di incasso MyBank, al fine di migliorare l'esperienza di pagamento del cliente e ottemperare a quanto previsto dalla disciplina in merito all'esposizione dei marchi, sarà necessario nella sezione "Metodo di pagamento" (o similare):

- denominare la specifica sezione collegata al link di pagamento in modo generico ad es."Paga" (o altro tasto di medesimo significato) e NON solo "Paga con carta di credito";
- mettere in evidenza i loghi "MyBank", "AccountToPay" e "Bancomat Pay", insieme ai loghi dei Circuiti di pagamento (es.Visa, MasterCard, …).

Quanto descritto sarà a valere anche per le altre soluzioni di incasso che tempo per tempo verranno introdotte.

2 Descrizione del processo

2.1 Registrazione utente sul sito/App dell'Esercente

I portali web, che offrono servizi di eCommerce, prevedono solitamente una fase iniziale di registrazione in cui richiedono al cliente dati di carattere anagrafico, uniti a riferimenti di posta elettronica e recapiti di domicilio e/o telefonici.
La fase di registrazione, permette di censire l'utente associandogli un profilo web corrispondente e, successivamente, di garantire un accesso, previa autenticazione, a tutela dei dati del cliente. Dopo essersi autenticato, il cliente può iniziare la navigazione sul sito/App eCommerce selezionando i prodotti di proprio interesse e popolando il carrello elettronico (shopping-cart) con la lista dei prodotti da ordinare.

2.2 Inizializzazione richiesta di pagamento

Ultimata la fase di selezione dei prodotti, il cliente viene veicolato su una pagina di conferma dell'acquisto. In questa pagina, dopo aver selezionato lo strumento da utilizzare per il pagamento, viene invitato a premere il pulsante relativo alla funzione di "acquisto/compra" o di uguale significato.

Ricevuta la conferma di acquisto, l'Esercente invia al payment gateway la richiesta pagamento, attraverso il metodo **`init()`** del servizio descritto nel wsdl **`PaymentInitGateway.wsdl o`** attraverso le API (classe **`IgfsCgInit`**).
Il messaggio contiene le seguenti proprietà obbligatorie:

- Identificativo dell'Esercente: **`tid`** (fornito dall'amministratore di PagOnline)
- Chiave per la firma digitale della richiesta: **`kSig`** (fornito dall'amministratore di PagOnline)
- Tipo di Transazione: **`trType`** (Pre Autorizzazione AUTH o Autorizzazione a livello contabile PURCHASE).
- Identificativo dell'ordine: **`shopID`**.
- Identificativo del cliente: **`shopUserRef`** (es. email dell'utente)
- Importo della Transazione: **`amount`**.
- Valuta della Transazione: **`currencyCode`**.
- Identificativo della lingua con la quale verrà visualizzata la pagina di pagamento: langID.
- URL http di errore dell'Esercente (pagina di errore definita dall'Esercente verso cui PagOnline deve eseguire la redirect del browser del cliente, a fronte di anomalie/errori durante il flusso di pagamento): **`errorUrl.`**
- URL http di notifica dell'Esercente (pagina di notifica definita dall'Esercente verso cui PagOnline deve eseguire una richiesta http/s informando l'Esercente che i dati di relativi all'esito della transazione di pagamento sono disponibili): **`notifyUrl`**

E' possibile inoltre inserire fino a 5 informazioni proprietarie attraverso le seguenti proprietà: addInfo1, addInfo2, addInfo3, addInfo4, addInfo5. Queste verranno memorizzate sulla base dati e

quindi legate all'ordine.

Al completamento della fase di inizializzazione, PagOnline restituisce all'Esercente le seguenti informazioni che si differenziano in base all'esito applicativo:

esito positivo:

- identificativo della richiesta di pagamento (proprietà **`paymentID`**), generato da PagOnline, che dovrà essere utilizzato nelle successive operazioni.
- URL per l'instradamento (proprietà **`redirectURL`**) verso la pagina per l'inserimento dei dati dello strumento di pagamento.

esito negativo:

- descrizione dell'errore rilevato da PagOnline.
  Ad esempio, in caso di invio di una richiesta con un campo mancante PagOnline risponde con l'esito **`IGFS_20000`** (proprietà **`rc`**) e la descrizione **`"Missing shopUserRef"`** (proprietà **`errorDesc`**).

  Per un elenco completo delle codifiche degli esiti previsti si veda *Appendice C.*

Utilizzando le API, le suddette informazioni sono fruibili utilizzando gli opportuni metodi "**`getter`**" della stessa.
Ad esempio la descrizione dell'errore, si può ottenere con l'invocazione del metodo **`getErrorDesc`**(), mentre l'URL per la redirect del browser del cliente sulla pagina "buy now", attraverso il metodo **`getRedirectURL`**().

Se l'inizializzazione è stata superata con successo l'Esercente effettua una redirect del browser Internet del cliente verso la pagina il cui URL è fruibile attraverso i Web Services.

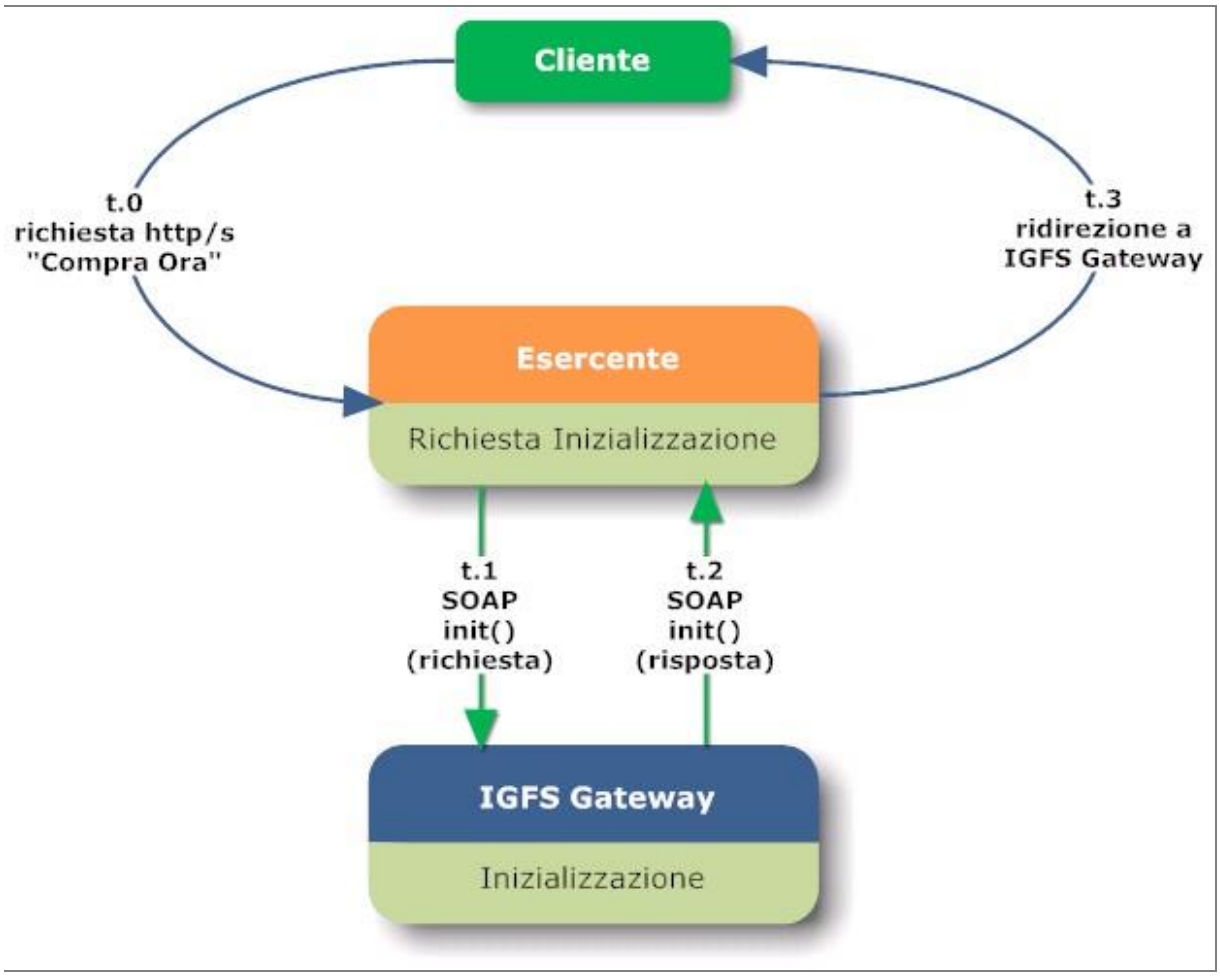


Figura 1 - Inizializzazione Richiesta di Pagamento

2.3 Pagina Strumenti di Pagamento

Il cliente, per completare l'operazione di pagamento, inserisce i dati richiesti nella pagina web proposta dal Payment Gateway in base allo strumento di pagamento selezionato.

Selezione tipologia di Pagamento

Nel caso venga utilizzato il codice TID che identifica il Selector al cliente sarà presentata una videata con gli strumenti di pagamento disponibili.
Solo dopo aver selezionato lo strumento di pagamento sarà visualizzata la videata specifica per l'inserimento dei dati.

Di seguito la videata proposta al cliente alla quale accede dopo aver premuto il tasto "Paga" o altro nome equivalente sul sito/App dell'Esercente che si avvale della modalità Selector (chiamate al payment gateway con TID Selector).

Figura 2 - Pagina di selezione dello strumento di pagamento

Pagamento con Carta di Credito

Figura 3 - Pagina di inserimento Dati di pagamento con Carta di Credito

Pagamento MyBank

Nel caso di pagamento MyBank sarà prima presentata la pagina per la scelta della banca

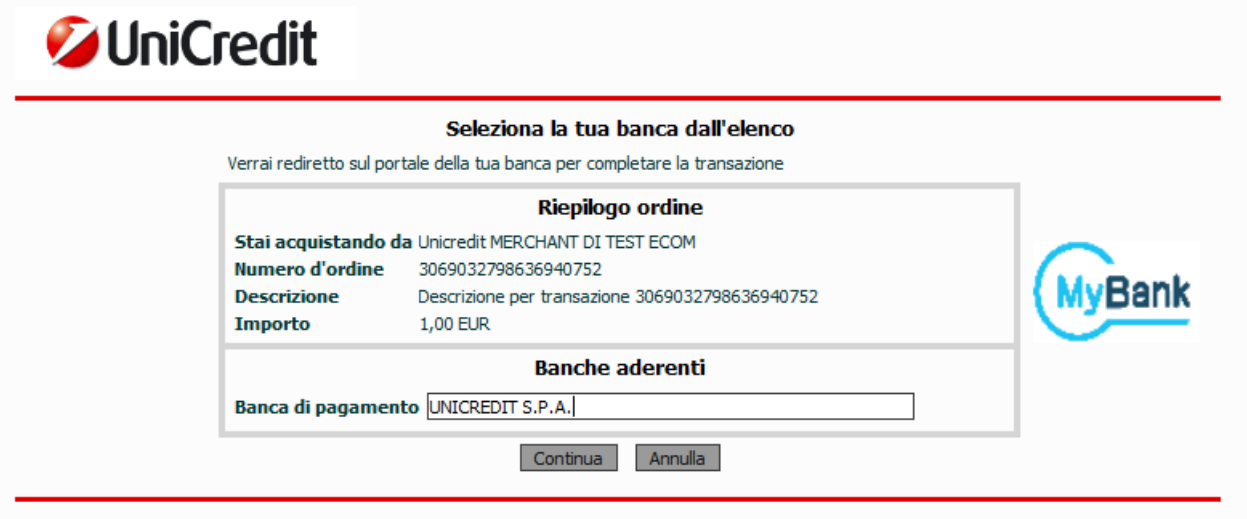


Figura 6 - MyBank - Pagina di scelta della banca

Effettuata la scelta, sarà presentata la videata specifica della Banca

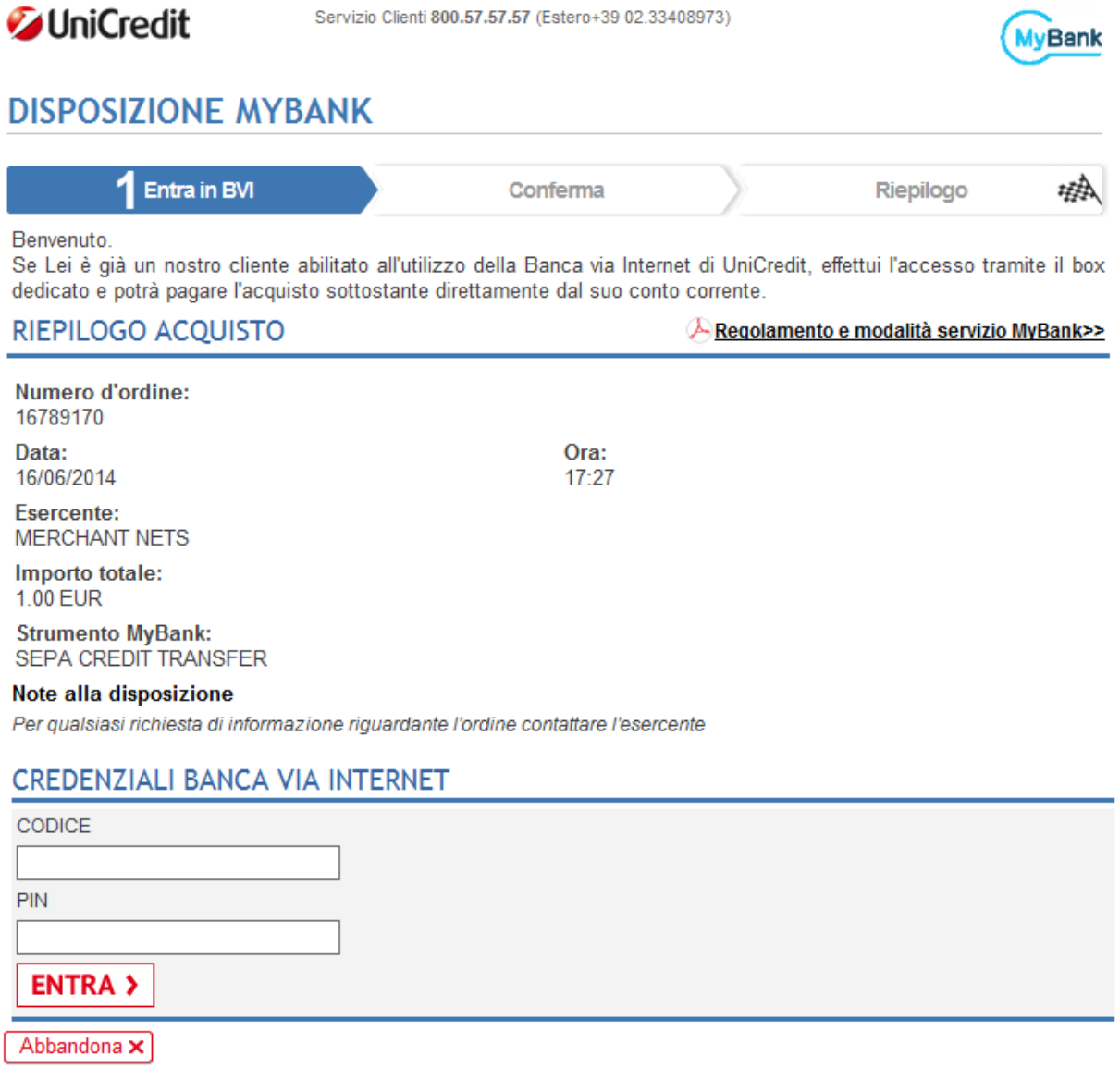


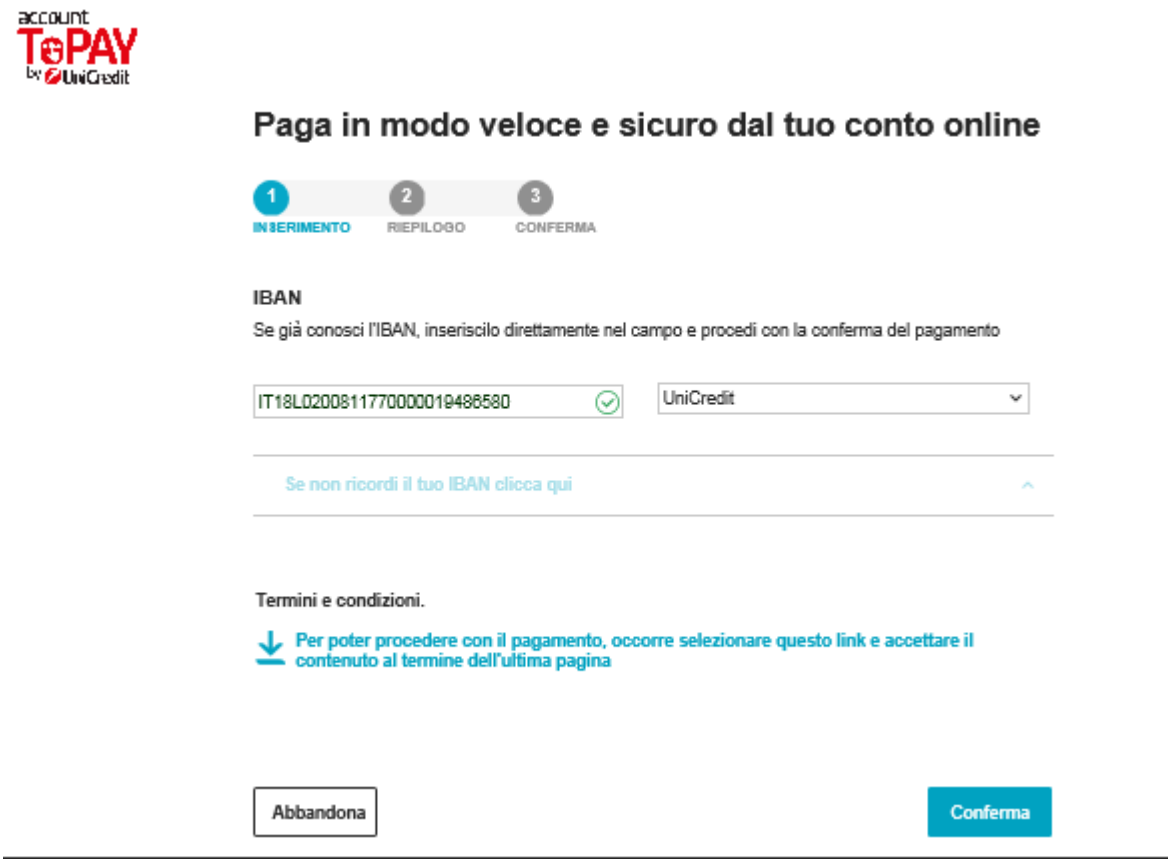
Figura 7 - MyBank - Pagina specifica della Banca

Pagamento AccountToPay

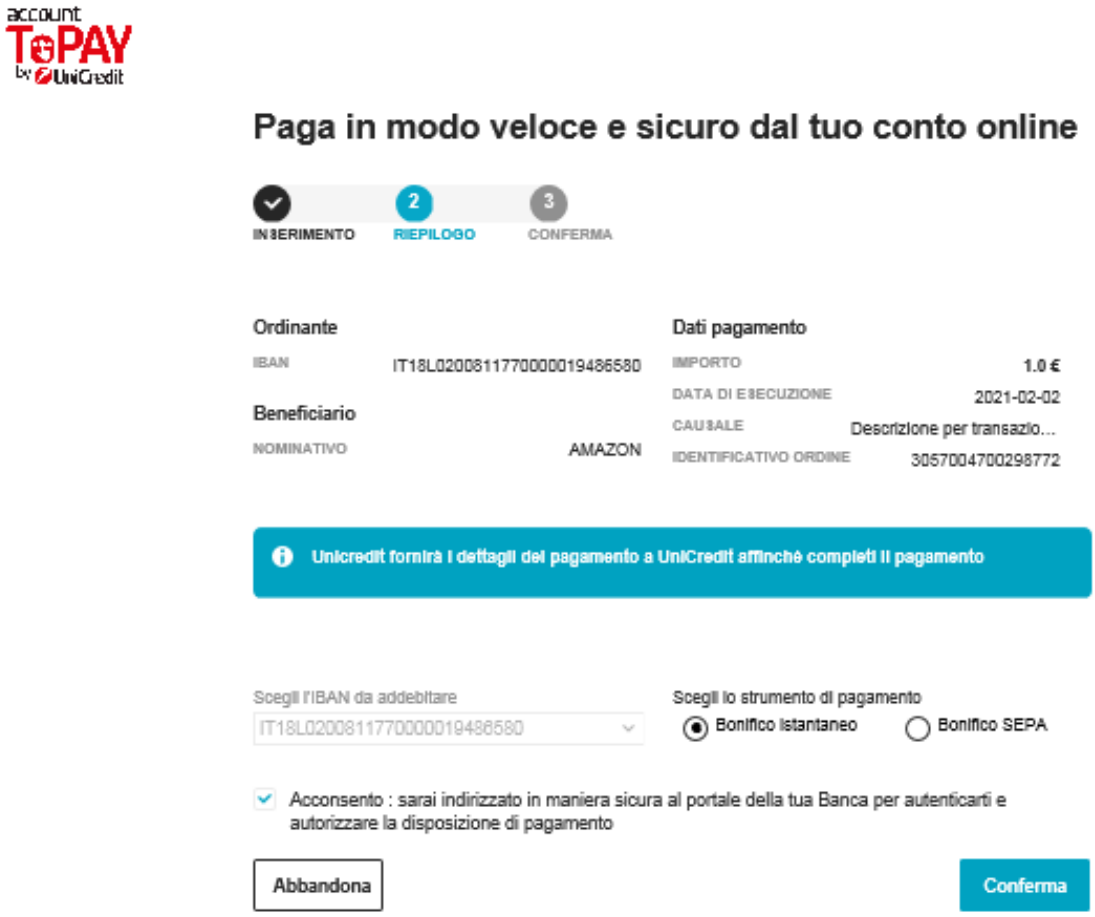
Nel caso di pagamento AccountToPay sarà prima presentata la pagina per l'inserimento dell'IBAN di addebito (accesso diretto tramite IBAN) o, nel caso il cliente non ricordi il proprio IBAN, scegliendo la propria Banca peril recupero dello stesso (avverrà nel contesto applicativo della specifica Banca di radicamento del conto).

Nella stessa pagina l'utente dovrà leggere e accettare i Termini e Condizioni di utilizzo del servizio

(aprendo il link).



Selezionando il tasto conferma si passa alla pagina di riepilogo dei dati dell'operazione, tutti precompilati e non modificabili dall'utente, che deve solo autorizzare l'inizializzazione del pagamento tramite reindirizzamento automatico alla propria banca (Processo PIS – Payment Initiation Service, previsto dalla normativa PSD2).



Pagamento Bancomat Pay

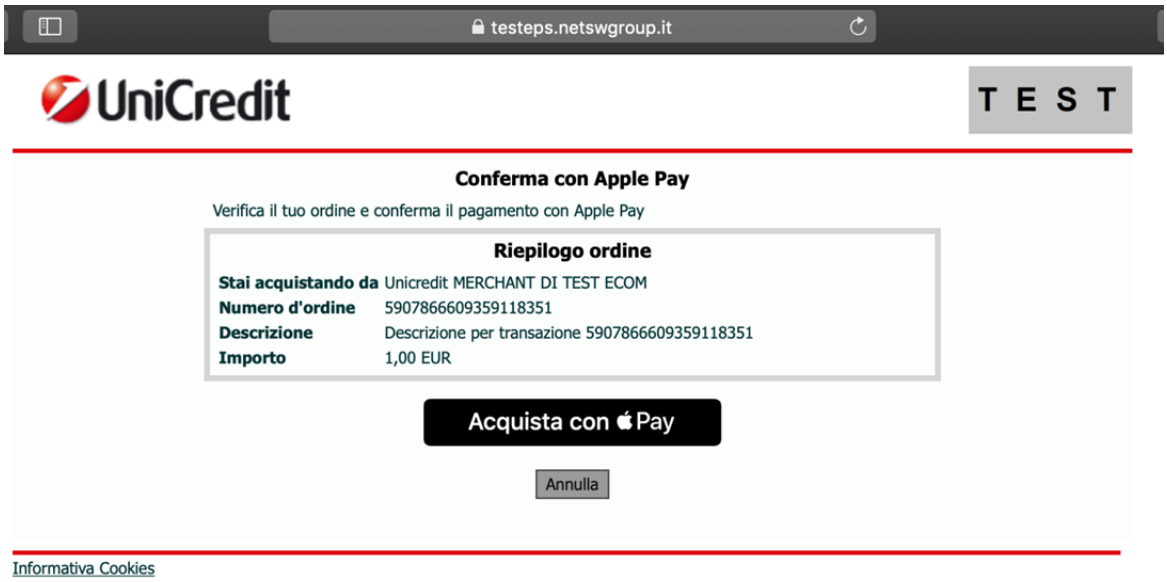
Nel caso di selezione del pagamento tramite Bancomat Pay, verrà visualizzata una schermata di richiesta del numero telefonico del pagatore; il pagatore dovrà inserire il numero di telefono inserito nella fase di registrazione sullo strumento di pagamento. Dopo aver digitato il numero di telefono l'utente potrà dare conferma (conferma: passo 1) al pagamento selezionando il bottone di conferma. Alla selezione della conferma il Payment Gateway inoltrerà la richiesta di pagamento alla piattaforma Bancomat Pay che, effettuate tutte le verifiche sul numero telefonico e lo stato del conto del pagatore (tutto in background), invierà una push notification all'applicazione mobile del pagatore. Il pagatore potrà confermare (conferma definitiva: passo 2) il pagamento attraverso la propria applicazione mobile. Nella figura sottostante un'estrazione della pagina di inserimento del numero telefonico.

Pagamento Alipay

Nel caso di pagamento tramite Alipay, verrà visualizzata una schermata con la pagina di login ed il QR Code da leggere con la applicazione dello strumento.
Il pagamento verrà successivamente confermato tramite App.

Pagamento Apple Pay

Il pagamento con Apple Pay potrà essere effettuato esclusivamente su prodotti a marchio Apple, con browser Safari su Desktop e con App Apple Pay su dispositivo mobile Apple (incluso AppleWatch)



2.4 Notifica all'Esercente

Al termine del pagamento, PagOnline ridirige il cliente verso l'URL di notifica che lo stesso Esercente ha comunicato a PagOnline in fase di inizializzazione.

A fronte di questa richiesta, per acquisire le informazioni relative all'esito della transazione di pagamento, l'Esercente deve invocare il metodo **`verify()`** del servizio descritto nel wsdl **`PaymentInitGateway.wsdl`**.

Nel caso in cui vengano utilizzate le API, questa operazione avviene attraverso la classe **`IgfsCgVerify`**

Ricevuti i dati relativi all'esito della transazione, l'Esercente dovrà redirigere il cliente verso la pagina di riepilogo.

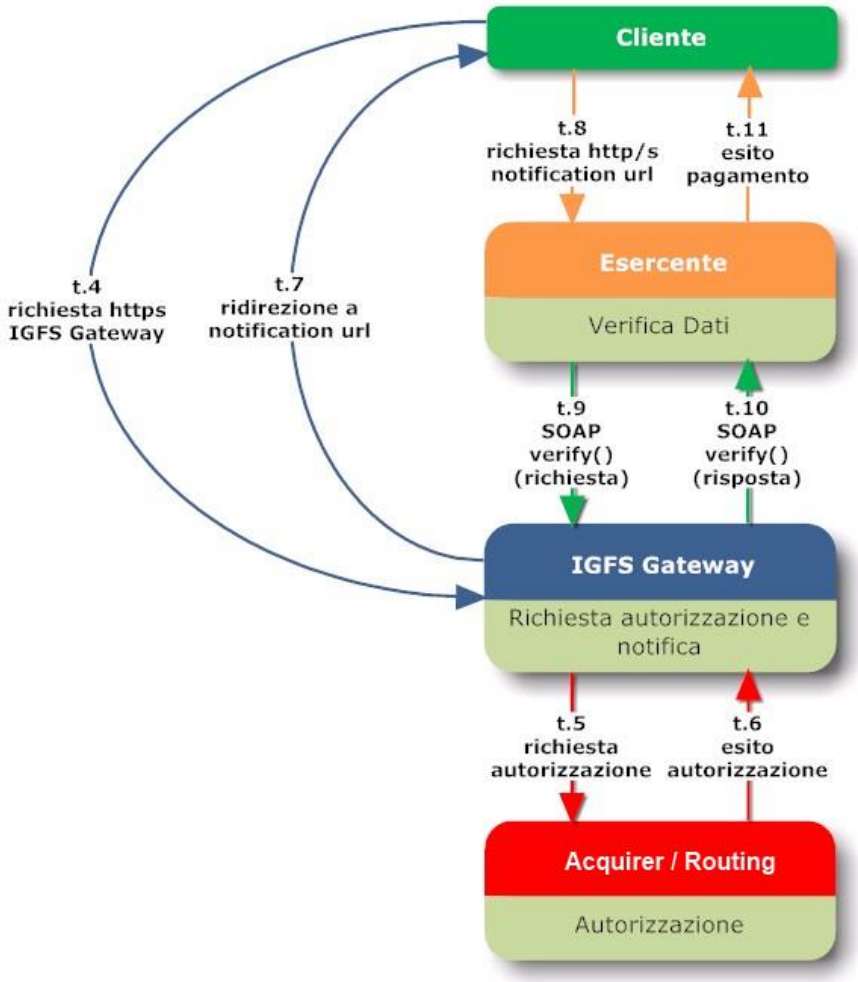


Figura 8 - Autorizzazione e Verifica Richiesta di Pagamento

Il flusso delle operazioni per la transazione di pagamento è terminato e il cliente può proseguire con la navigazione sul portale dell'Esercente.

2.5 Funzionalità batch

Il perfezionamento dell'acquisto (addebito al titolare), il suo annullamento (ripristino del plafond di spesa) o la restituzione degli importi addebitati per acquisti già perfezionati (accredito al titolare) può essere effettuato anche in modalità batch.

Tale modalità, consigliata per richieste massive, prevede:

- la sottomissione a PagOnline di un file contenente le informazioni necessarie per l'individuazione delle transazioni e delle operazioni da effettuare su di esse;
- l'attesa del completamento della elaborazione del file sottomesso;
- il recupero del file contenente gli esiti delle operazioni richieste.

Nel caso in cui le operazioni online siano state effettuate specificando l'addebito contestuale all'autorizzazione (PURCHASE) l'unica operazione permessa è il rimborso (accredito al titolare).

Il formato dei file da inviare e da ricevere vengono descritti nell'*Appendice B.*

2.6 Invio file

La richiesta di invio del file viene effettuata al PagOnline Imprese attraverso il metodo **`submit()`** del servizio descritto nel wsdl **`BatchGateway.wsdl o`** attraverso le API (classe **`IgfsCgBatchSubmit`**).

Il messaggio contiene le seguenti proprietà obbligatorie:

- Identificativo dell'Esercente: **`tid`**. (fornito dall'amministratore di PagOnline Imprese)
- Chiave per la firma digitale della richiesta: **`kSig`** (fornito dall'amministratore di PagOnline)
- Identificativo batch per l'Esercente: **`BatchShopID`**
- Contenuto File da inviare: **`BatchData`**

A completamento della fase di acquisizione file, PagOnline Imprese risponde all'Esercente con informazioni che si differenziano in base all'esito applicativo:

esito positivo:

- Identificativo della richiesta di elaborazione batch (proprietà **`batchID`**), generato da PagOnline Imprese.

esito negativo:

- Descrizione dell'errore rilevato da PagOnline Imprese.
  Per un elenco completo delle codifiche degli esiti previsti si veda *Appendice C.*

2.6.1 Acquisizione file esiti

La richiesta di acquisizione del file con gli esiti viene effettuata al gateway PagOnline Imprese attraverso il metodo **`fetch()`** del servizio descritto nel wsdl **`BatchGateway.wsdl o`** attraverso le API (classe **`IgfsBatchFetch`**).

Il messaggio contiene le seguenti proprietà obbligatorie:

- Identificativo dell'Esercente: **`tid`**.
- Chiave per la firma digitale della richiesta : **`kSig`**
- Identificativo batch per il quale si richiede l'esito: **`BatchShopID`**

A completamento della fase di acquisizione file, PagOnline Imprese risponde all'Esercente con informazioni che si differenziano in base all'esito applicativo:

esito positivo:

- Stato della richiesta di elaborazione batch (proprietà **`status`**);
  Può assumere uno dei seguenti valori:
  - *NOT_PROCESSED*
  - *PROCESSING*
  - *PROCESSED*
  - *ERROR*

esito negativo:

- descrizione dell'errore rilevato da PagOnline Imprese.
  Per un elenco completo delle codifiche degli esiti previsti si veda *Allegato A.*

Nello stato **`NOT_PROCESSED`** o **`PROCESSING`** l'elaborazione non risulta ancora completata ed è quindi necessario attendere e procedere a una successive interrogazione sino al completamento con successo (**`PROCESSED`**) o con errori (**`ERROR`**).

In caso di completamento positivo le proprietà **`size`** e **`batchData`** conterranno, rispettivamente, le dimensioni e il contenuto del file esiti.

3 Web services

3.1 PaymentInitGateway.wsdl (Pagamenti online)

I metodi sotto descritti sono da utilizzare quando avviene l'integrazione per strumenti eCommerce in cui l'utente effettua direttamente l'inserimento dei dati per poter completare il pagamento.
Gli strumenti di pagamento eCommerce sono:

- Carte di Credito;
- MyBank (se convenzionato);
- AccountToPay
- Bancomat Pay

altri strumenti di pagamento tempo per tempo abilitati dalla Banca.

3.1.1 Metodo init()

Il metodo:

```
PaymentInitResponse init(PaymentInitRequest request);
```

del servizio **`PaymentInitGateway`** viene utilizzato per eseguire una inizializzazione della richiesta di pagamento.

Di seguito l'elenco delle proprietà della richiesta e della risposta.

PaymentInitRequest		
Tipo[Dimensione]	**Dato**	**Proprietà**

String		**Signature** Firma del messaggio composta dalla concatenazione dei campi: - Tid - ShopID - ShopUserRef - ShopUserName - ShopUserAccount - TrType - Amount - CurrencyCode - LangID - NotifyURL - ErrorURL - AddInfo1 - AddInfo2 - AddInfo3 - AddInfo4 - AddInfo5 - Description - Recurrent - PaymentReason - FreeText - ValidityExpire Per il calcolo della firma si veda l'*APPENDICE A.*
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**ShopID** Chiave esterna identificante il pagamento
String[256]	Opzionale	**ShopUserRef** Identificativo cliente (es:email)
String[256]	Opzionale	**ShopUserName** Cognome e Nome del cliente (separati dal carattere ,) (es. rossi,mario)
String[64]	Opzionale	**ShopUserAccount** Account cliente del portale Esercente
PURCHASE, AUTH, VERIFY	Obbligatorio	**TrType** Tipologia della richiesta AUTH = Pre Autorizzazione PURCHASE = Autorizzazione a livello contabile VERIFY = Verifica stato carta (nessun addebito o blocco plafond)

Long[12]	Condizionale	**Amount** Importo in virgola virtuale (es. 100 = 1,00 EUR) Obbligatorio con **TrType** PURCHASE o AUTH Opzionale con **TrType** VERIFY
EUR	Condizionale	**CurrencyCode** Valuta Obbligatorio con **TrType** PURCHASE o AUTH Opzionale con **TrType** VERIFY
IT, EN	Obbligatorio	**LangID** Codice iso 639-2 relativo alla pagina di inserimento dei dati di pagamento
URL[512]	Obbligatorio	**NotifyURL** URL relativo alla pagina di notifica esito
URL[512]	Obbligatorio	**ErrorURL** URL relativo alla pagina di errore
String[256]	Opzionale	**AddInfo1** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo2** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo3** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo4** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo5** Campo a disposizione dell'Esercente
String[100]	Opzionale	**Description** Causale di pagamento
Boolean	Opzionale	**Recurrent** Pagamento ricorrente
String[99]	Opzionale	**PaymentReason** Ragione del pagamento
String[268]	Opzionale	**FreeText** Testo libero
DateTime	Opzionale	**ValidityExpire** Tempo limite consentito per la validità della transazione (valido solo per strumenti di pagamento esterni)

PaymentInitResponse	
Tipo[Dimensione]	**Proprietà**
Boolean	**Error** Restituisce true in presenza di un errore/anomalia
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione dell'errore/anomalia
String[32]	**PaymentID** Codice paymentID associato alla richiesta
URL[512]	**RedirectURL** Url associato alla pagina di "buynow"

3.1.2 Metodo verify()

Il metodo:

`PaymentVerifyResponse verify(PaymentVerifyRequest request);`

del servizio **`PaymentInitGateway`** viene utilizzato per eseguire una operazione di verifica dati della richiesta di pagamento.

Di seguito l'elenco delle proprietà della richiesta e della risposta.

PaymentVerifyRequest		
Tipo[Dimensione]	**Dato**	**Proprietà**
String	Obbligatorio	**Signature** Firma del messaggio composta dalla concatenazione dei campi: - Tid - ShopID - PaymentID Per il calcolo della firma si veda l'*APPENDICE A.*
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**ShopID** Chiave esterna identificante il pagamento
String[32]	Obbligatorio	**PaymentID** Codice paymentID associato alla richiesta

PaymentVerifyResponse	
Tipo[Dimensione]	**Proprietà**
Boolean	**Error** Restituisce true in presenza di un errore/anomalia
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione di un errore/anomalia
Long[16]	**TranID** Codice Ordine processato
String[32]	**AuthCode** Codice di autorizzazione restituito dall' issuer
String[1]	**EnrStatus** Stato di iscrizione carta al servizio 3D Secure
String[1]	**AuthStatus** Esito autenticazione carta al servizio 3D Secure
String[32]	**Brand** Brand carta di credito es. (VISA, MASTERCARD,…)

3.1.3 Metodo Selector()

Il metodo:

`PaymentSelectorResponse selector(PaymentSelectorRequest request);`

del servizio **`PaymentInitGateway`** viene utilizzato per ottenere la lista dei possibili strumenti di pagamento legati al selettore richiesto.

Di seguito l'elenco delle proprietà della richiesta e della risposta.

PaymentSelectorRequest		
Tipo[Dimensione]	**Dato**	**Proprietà**
String	Obbligatorio	**Signature** Firma del messaggio composta dalla concatenazione dei campi: - Tid - ShopID - ShopUserRef - TrType - Amount - CurrencyCode - LangID - AddInfo1 - AddInfo2 - AddInfo3 - AddInfo4 - AddInfo5 Per il calcolo della firma si veda l'*APPENDICE A.*
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**ShopID** Chiave esterna identificante il pagamento
String[256]	Opzionale	**ShopUserRef** Identificativo cliente (es:email)
PURCHASE, AUTH, VERIFY	Obbligatorio	**TrType** Tipologia della richiesta AUTH = Pre Autorizzazione PURCHASE = Autorizzazione a livello contabile VERIFY = Verifica stato carta (nessun addebito o blocco plafond)

Long[12]	Condizionale	**Amount** Importo in virgola virtuale (es. 100 = 1,00 EUR) Obbligatorio con **TrType** PURCHASE o AUTH Opzionale con **TrType** VERIFY
EUR	Condizionale	**CurrencyCode** Valuta Obbligatorio con **TrType** PURCHASE o AUTH Opzionale con **TrType** VERIFY
IT, EN	Obbligatorio	**LangID** Codice iso 639-2 relativo alla pagina di inserimento dei dati di pagamento
String[256]	Opzionale	**AddInfo1** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo2** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo3** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo4** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo5** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo5** Campo a disposizione dell'Esercente

PaymentSelectorResponse	
Tipo[Dimensione]	**Proprietà**
Boolean	**Error** Restituisce true in presenza di un errore/anomalia
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione dell'errore/anomalia
String[32]	**PaymentID** Codice paymentID associato alla richiesta
TerminalInfo[]	**Terminal** Terminali abilitati al pagamento (0..*)

TerminalInfo	
Tipo[Dimensione]	**Proprietà**
String[16]	**Tid** Codice terminale dell'Esercente
String[256]	**Description** Descrizione codice terminale dell'Esercente
String[2]	**PayInstr** Codice della modalità di pagamento
String[64]	**PayInstrDescription** Descrizione della modalità di pagamento (Label del bottone)
URL[512][]	**ImgUrl** Url associato all'immagine (0..*)

3.2 PaymentTranGateway.wsdl (Servizi diretti)

3.2.1 Metodo confirm()

Il metodo:

```
PaymentConfirmResponse confirm(PaymentConfirmRequest request);
```

del servizio **PaymentTranGateway** viene utilizzato per movimentare una autorizzazione effettata con carta di credito.

Di seguito l'elenco delle proprietà della richiesta e della risposta.

PaymentConfirmRequest		
Tipo[Dimensione]	**Dato**	**Proprietà**
String	Obbligatorio	**Signature** Firma del messaggio composta dalla concatenazione dei campi: - Tid - ShopID - Amount - RefTranID - SplitTran - AddInfo1 - AddInfo2 - AddInfo3 - AddInfo4 - AddInfo5 Per il calcolo della firma si veda l'*APPENDICE A.*
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**ShopID** Chiave esterna identificante il pagamento
Long[12]	Obbligatorio	**Amount** Importo in virgola virtuale (es. 100 = 1,00 EUR)
Long[16]	Obbligatorio	**RefTranID** Codice Ordine relativo alla transazione da movimentare
Boolean	Opzionale	**SplitTran** è true se la conferma è parziale

String[256]	Opzionale	**AddInfo1** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo2** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo3** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo4** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo5** Campo a disposizione dell'Esercente

PaymentConfirmResponse	
Tipo[Dimensione]	**Proprietà**
Boolean	**Error** Restituisce true in presenza di un errore/anomalia
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione dell'errore/anomalia
Long[16]	**TranID** Codice Ordine processato
String[256]	**AddInfo1** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo2** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo3** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo4** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo5** Dati inviati in fase di autorizzazione dall'Esercente
Long[12]	**PendingAmount** Eventuale importo non confermato

3.2.2 Metodo voidAuth()

Il metodo:

`PaymentVoidAuthResponse voidAuth(PaymentVoidAuthRequest request);`

del servizio **`PaymentTranGateway`** viene utilizzato per stornare una autorizzazione.

Di seguito l'elenco delle proprietà della richiesta e della risposta.

PaymentVoidAuthRequest		
Tipo[Dimensione]	**Dato**	**Proprietà**
String	Obbligatorio	**Signature** Firma del messaggio composta dalla concatenazione dei campi: - Tid - ShopID - Amount - RefTranID - AddInfo1 - AddInfo2 - AddInfo3 - AddInfo4 - AddInfo5 Per il calcolo della firma si veda l'*APPENDICE A.*
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**ShopID** Chiave esterna identificante il pagamento
Long[12]	Obbligatorio	**Amount** Importo in virgola virtuale (es. 100 = 1,00 EUR)
Long[16]	Obbligatorio	**RefTranID** Codice Ordine relativo alla transazione da annullare
String[256]	Opzionale	**AddInfo1** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo2** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo3** Campo a disposizione dell'Esercente

String[256]	Opzionale	**AddInfo4** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo5** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo5** Campo a disposizione dell'Esercente

PaymentVoidAuthResponse	
Tipo[Dimensione]	**Proprietà**
Boolean	**Error** Restituisce true in presenza di un errore/anomalia
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione dell'errore/anomalia
Long[16]	**TranID** Codice Ordine processato
String[256]	**AddInfo1** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo2** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo3** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo4** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo5** Dati inviati in fase di autorizzazione dall'Esercente

3.2.3 Metodo credit()

Il metodo:

`PaymentCreditResponse credit(PaymentCreditRequest request);`

del servizio **`PaymentTranGateway`** viene utilizzato per riaccreditare una movimentazione.

Di seguito l'elenco delle proprietà della richiesta e della risposta.

PaymentCreditRequest		
Tipo[Dimensione]	**Dato**	**Proprietà**
String	Obbligatorio	**Signature** Firma del messaggio composta dalla concatenazione dei campi: - Tid - ShopID - Amount - RefTranID - SplitTran - AddInfo1 - AddInfo2 - AddInfo3 - AddInfo4 - AddInfo5 Per il calcolo della firma si veda l'*APPENDICE A.*
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**ShopID** **Chiave esterna identificante il pagamento**
Long[12]	Obbligatorio	**Amount** Importo in virgola virtuale (es. 100 = 1,00 EUR)
Long[16]	Obbligatorio	**RefTranID** Codice Ordine relativo alla transazione da riaccreditare
Boolean	Opzionale	**SplitTran** è true se la conferma è parziale
String[256]	Opzionale	**AddInfo1** Campo a disposizione dell'Esercente

String[256]	Opzionale	**AddInfo2** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo3** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo4** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo5** Campo a disposizione dell'Esercente
String[256]	Opzionale	**AddInfo5** Campo a disposizione dell'Esercente

I rimborsi possono essere effettuati tramite BackOffice entro i seguenti termini:

- Visa e Mastercard: entro **6 mesi** dalla data della operazione originaria o senza limiti di tempo tramite API
- Bancomat Pay: entro 30 giorni dalla data della operazione originaria
- Alipay: entro 12 mesi dalla data operazione originaria

PaymentCreditResponse	
Tipo[Dimensione]	**Proprietà**
Boolean	**Error** Restituisce true in presenza di un errore/anomalia
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione dell'errore/anomalia
Long[16]	**TranID** Codice Ordine processato
String[256]	**AddInfo1** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo2** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo3** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo4** Dati inviati in fase di autorizzazione dall'Esercente

String[256]	**AddInfo5** Dati inviati in fase di autorizzazione dall'Esercente

3.3 BatchGateway.wsdl (Funzionalità batch)

Il modulo batch è basato sulle specifiche MTOM (http://www.w3.org/TR/soap12-mtom/) pertanto è necessario utilizzare un Client SOAP compatibile con tale specifica (Apache Axis 2, Apache CXF, Microsoft .Net); nel caso in cui ciò non fosse possibile si raccomanda l'utilizzo delle API.

Per il formato del file di batch fare riferimento all'**APPENDICE B: FILE BATCH** .

3.3.1 Metodo submit()

Il metodo:

```
BatchSubmitResponse submit(BatchSubmitRequest request);
```

del servizio **BatchGateway** viene utilizzato per inviare il file di conferma, ovvero le azione da intraprendere sulle transazioni specificate (conferma, storno, credito).

Di seguito l'elenco delle proprietà della richiesta e della risposta.

	BatchSubmitRequest	
Tipo[Dimensione]	**Dato**	**Proprietà**
String	Obbligatorio	**Signature** Firma del messaggio composta dalla concatenazione dei campi: - Tid - BatchShopID - BatchData Per il calcolo della firma si veda l'*APPENDICE A.*
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**BatchShopID** Codice Batch associato alla richiesta
Byte[]	Obbligatorio	**BatchData** Contenuto File conferme

BatchSubmitResponse	
Tipo[Dimensione]	**Proprietà**
Boolean	**Error** Restituisce true in presenza di un errore/anomalia
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione di un errore/anomalia
String[18]	**BatchID** Identificatore Batch

3.3.2 Metodo fetch()

Il metodo:

`BatchFetchResponse fetch(BatchFetchRequest request);`

del servizio **`BatchGateway`** viene utilizzato per prelevare il file di risposta al file conferme inviato tramite il metodo **`submit()`**.

Di seguito l'elenco delle proprietà della richiesta e della risposta.

BatchFetchRequest		
Tipo[Dimensione]	**Dato**	**Proprietà**
String	Obbligatorio	**Signature** Firma del messaggio composta dalla concatenazione dei campi: - Tid - BatchShopID Per il calcolo della firma si veda l'*APPENDICE A*.
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**BatchShopID** Codice Batch associato alla richiesta

BatchFetchResponse	
Tipo[Dimensione]	**Proprietà**
Boolean	**Error** Restituisce true in presenza di un errore/anomalia
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione di un errore/anomalia
String[32]	**Status** Stato elaborazione Batch Possibili valori: *NOT_PROCESSED* *PROCESSING* *PROCESSED* *ERROR*
Long	**Size** Dimensione del file risposta
Byte[]	**BatchData** Contenuto File esiti

3.4 Esempio di implementazione Pagamento online

Al fine di facilitare la comprensione dei metodi relativi all'utilizzo dei Web Services sopra descritti, riportiamo di seguito, a titolo di esempio, le operazioni applicative necessarie per la sottomissione di una richiesta di pagamento verso il gateway PagOnline.

Un ambiente di test è disponibile ai seguenti URL:

- https://testeps.netswgroup.it/UNI_CG_SERVICES/services/PaymentInitGatewayPort?wsdl
  (URL se si utilizzano i WSDL)
- https://testeps.netswgroup.it/UNI_CG_SERVICES/services
  (URL se si utilizzano le API)

Tali esempi sono basati sul framework Apache CXF e sulle classi client da esso generate.

3.4.1 WS Init

```
<%
// ====================================================================
// =          importazione classi di riferimento                      =
// ====================================================================
%>
<%@page import="import java.io.File" %>
<%@page import="import java.net.MalformedURLException" %>
<%@page import="import java.net.URL" %>
<%@page import="import javax.xml.namespace.QName" %>
<%

// ====================================================================
// = impostazione parametri per l'inizializzazione richiesta di       =
// = pagamento.                                                       =
// = NB: I parametri riportati sono solo a titolo di esempio          =
// ====================================================================
URL wsdlURL = new URL("https://
testeps.netswgroup.it/UNI_CG_SERVICES/services/PaymentInitGatewayPort?wsdl");
QName SERVICE_NAME = new QName("http://services.api.web.cg.igfs.apps.netsw.it/",
"PaymentInitGateway");

PaymentInitGateway_Service ss = new PaymentInitGateway_Service(wsdlURL,
SERVICE_NAME);
PaymentInitGateway port = ss.getPaymentInitGatewayPort();

String tid           = "UNI_ECOM"; //per servizio MyBank usare UNI_MYBK
                                   //per AccountToPay utilizzare UNI_UNIPAY
                                   //per BancomatPat utilizzare UNI_BPAY

String kSig          = "UNI_TESTKEY";
String shopID        = "56870108202724855455"; // Chiave esterna UNIVOCA
identificante il pagamento
String email         = "user@customer.it";
```

```
String trType         = "AUTH";
long amount           = 100;
String curCode        = "EUR";
String langID         = "IT";
String errorURL       = "https://Esercente/error.jsp";
String notifyURL      = "https://Esercente/notify.jsp";

String signature = getSignature(kSig, // KSIGN
                                tid, // TID
                                shopID, // SHOPID
                                shopUserRef, // SHOPUSERREF
                                trType,// TRTYPE
                                amount, // AMOUNT
                                currencyCode, // CURRENCYCODE
                                langID, // LANGID
                                notifyURL, // NOTIFYURL
                                errorURL); // ERRORURL



Init _init_parameters = new Init();
PaymentInitRequest _init_parametersRequest = new  PaymentInitRequest();
_init_parametersRequest.setTid(tid);
_init_parametersRequest.setSignature(signature);
_init_parametersRequest.setShopID(shopID);
_init_parametersRequest.setShopUserRef(shopUserRef);
_init_parametersRequest.setTrType(trType);
_init_parametersRequest.setAmount(amount);
_init_parametersRequest.setCurrencyCode(currencyCode);
_init_parametersRequest.setLangID(langID);
_init_parametersRequest.setNotifyURL(notifyURL);
_init_parametersRequest.setErrorURL(errorURL);
_init_parameters.setRequest(_init_parametersRequest);

// ====================================================================
// =              esecuzione richiesta di inizializzazione            =
// ====================================================================
InitResponse _init_return = port.init(_init_parameters);

if (_init_return.getResponse().isError()) {
  // ================================================================
  // = redirect del client su pagina di errore definita dall'Esercente  =
  // ================================================================
  response.sendRedirect(errorURL + "?rc=" + _init_return.getResponse().getRc() +
                        "&errorDesc=" + _init_return.getResponse().getErrorDesc());
  return;
}

String paymentID = _init_return.getResponse().getPaymentID();
// NOTA: Salvo il paymentID relativo alla richiesta (es. sul DB)...

// ====================================================================
// =          redirect del client verso URL PagOnline Imprese         =
// ====================================================================
String redirectURL = _init_return.getResponse().getRedirectURL();
response.sendRedirect(redirectURL.toString());
%>
```

3.4.2 WS Verify

```
<%
// ====================================================================
// =          importazione classi di riferimento                      =
// ====================================================================
%>
<%@page import="import java.io.File" %>
<%@page import="import java.net.MalformedURLException" %>
<%@page import="import java.net.URL" %>
<%@page import="import javax.xml.namespace.QName" %>
<%

// ====================================================================
// = impostazione parametri per l'inizializzazione richiesta di       =
// = pagamento.                                                       =
// = NB: I parametri riportati sono solo a titolo di esempio          =
// ====================================================================
URL wsdlURL = new URL("https://testeps.netswgroup.it/UNI_CG_SERVICES
/services/PaymentInitGatewayPort?wsdl");
QName SERVICE_NAME = new QName("http://services.api.web.cg.igfs.apps.netsw.it/",
"PaymentInitGateway");

PaymentInitGateway_Service ss = new PaymentInitGateway_Service(wsdlURL,
SERVICE_NAME);
PaymentInitGateway port = ss.getPaymentInitGatewayPort();

String tid           = "UNI_ECOM"; //per servizio MyBank usare UNI_MYBK
String kSig          = "UNI_TESTKEY";
String shopID        = "5687010820272485455"; // Chiave esterna UNIVOCA
                        identificante il pagamento
String paymentID     = // NOTA: Leggo il paymentID rilasciato in fase di init
                        (es. dal DB)...
String errorURL      = "https://Esercente/error.jsp";
String esitoURL      = "https://Esercente/esito.jsp";

String signature = getSignature(kSig, // KSIGN
                                tid, // TID
                                shopID, // SHOPID
                                paymentID); // PAYMENTID

Verify _verify_parameters = new Verify();
PaymentVerifyRequest _verify_parametersRequest = new PaymentVerifyRequest();
_verify_parametersRequest.setTid(tid);
_verify_parametersRequest.setSignature(signature);
_verify_parametersRequest.setShopID(shopID);
_verify_parametersRequest.setPaymentID(paymentID);
_verify_parameters.setRequest(_verify_parametersRequest);


// ====================================================================
// =              esecuzione richiesta di verifica                    =
// ====================================================================
VerifyResponse _verify_return = port.verify(_verify_parameters);
```

```
if (_verify_return.getResponse().isError()) {
  // ========================================================================
  // = redirect del client su pagina di errore definita dall'Esercente  =
  // ========================================================================
  response.sendRedirect(errorURL + "?rc=" +
                        _verify_return.getResponse().getRc() + "&errorDesc=" +
                        _verify_return.getResponse().getErrorDesc());
  return;
}

// ========================================================================
// =      redirect del client verso URL Esito Pagamento Esercente      =
// ========================================================================
StringBuffer resultUrl = new StringBuffer();
resultUrl.append(esitoURL);
resultUrl.append("?rc=" + verify.getRc());
resultUrl.append("&tranID=" + verify.getTranID());
resultUrl.append("&enrStatus=" + verify.getEnrStatus());
resultUrl.append("&authStatus=" + verify.getAuthStatus());
response.sendRedirect(resultUrl.toString());
%>
```

4 API

4.1 Pagamenti online

4.1.1 Classe IgfsCgInit

La classe **`IgfsCgInit`** viene utilizzata per eseguire una inizializzazione della richiesta di pagamento.

Sommario Properties Input		
Tipo[Dimensione]	**Dato**	**Proprietà**
URL	Obbligatorio	**ServerURL** Indirizzo del server di destinazione della richiesta
Integer	Opzionale	**Timeout** Timeout massimo espresso in millisecondi di completamento di una richiesta
String[64]	Obbligatorio	**KSig** Chiave per firmare il messaggio
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**ShopID** Chiave esterna identificante il pagamento
String[256]	Opzionale	**ShopUserRef** Identificativo chiente
String[256]	Opzionale	**ShopUserName** Cognome e Nome del cliente (separati dal carattere ,) (es. rossi,mario)
String[64]	Opzionale	**ShopUserAccount** Account cliente del portale Esercente
PURCHASE, AUTH, VERIFY	Obbligatorio	**TrType** Tipologia di una richiesta AUTH = Pre Autorizzazione PURCHASE = Autorizzazione a livello contabile VERIFY = Verifica stato carta (nessun addebito o blocco plafond)

Long[12]	Condizionale	**Amount** Importo in virgola virtuale (es. 100 = 1,00 EUR) Obbligatorio con **TrType** PURCHASE o AUTH Opzionale con **TrType** VERIFY
EUR	Condizionale	**CurrencyCode** Valuta Obbligatorio con **TrType** PURCHASE o AUTH Opzionale con **TrType** VERIFY
IT, EN	Obbligatorio	**LangID** Lingua relativa alla pagina di inserimento dei dati sensibili associata ad una richiesta
URL[512]	Obbligatorio	**NotifyURL** URL relativo alla pagina di notifica esito di una richiesta
URL[512]	Obbligatorio	**ErrorURL** URL relativo alla pagina di errore associata ad una richiesta
String[256]	Opzionale	**AddInfo1** Campo a disposizione dell' Esercente
String[256]	Opzionale	**AddInfo2** Campo a disposizione dell' Esercente
String[256]	Opzionale	**AddInfo3** Campo a disposizione dell' Esercente
String[256]	Opzionale	**AddInfo4** Campo a disposizione dell' Esercente
String[256]	Opzionale	**AddInfo5** Campo a disposizione dell' Esercente
String[100]	Opzionale	**Description** Descrizione del pagamento
String[99]	Opzionale	**PaymentReason** Causale di pagamento
Boolean	Opzionale	**Recurrent** Pagamento ricorrente
String[256]	Opzionale	**AccountName** Cognome e Nome del titolare (separato da ,) (es. rossi,mario)
DateTime	Opzionale	**ValidityExpire** Tempo di validità della transazione (valido solamente per strumenti di pagamento esterni)

Metodi	
Tipo[Dimensione]	**Proprietà**
Boolean	**execute**() Esegue la transazione
	resetFields() Azzeramento parametri di richiesta

Sommario Properties Output	
Tipo[Dimensione]	**Proprietà**
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione di un errore/anomalia
String[32]	**PaymentID** Codice paymentID associato ad una richiesta
URL[512]	**RedirectURL** Url associato alla pagina di PagOnline Imprese

4.1.2 Classe IgfsCgVerify

La classe **`IgfsCgVerify`** viene utilizzata per eseguire una operazione di verifica dati della richiesta di pagamento.

Sommario Properties Input		
Tipo[Dimensione]	**Dato**	**Proprietà**
URL	Obbligatorio	**ServerURL** Indirizzo del server di destinazione della richiesta
Integer	Opzionale	**Timeout** Timeout massimo espresso in millisecondi di completamento di una richiesta
String[64]	Obbligatorio	**KSig** Chiave per firmare il messaggio
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**ShopID** Chiave esterna identificante il pagamento
String[256]	Obbligatorio	**PaymentID** Codice paymentID associato ad una richiesta

Metodi	
Tipo[Dimensione]	**Proprietà**
Boolean	**execute**() Esegue la transazione
	resetFields() Azzeramento parametri di richiesta

Sommario Properties Output	
Tipo[Dimensione]	**Proprietà**
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione di un errore/anomalia

Long[16]	**TranID** Codice Ordine processato
String[32]	**AuthCode** Codice di autorizzazione restituito dall' issuer
String[1]	**EnrStatus** Stato di iscrizione carta al servizio 3D Secure
String[1]	**AuthStatus** Esito autenticazione carta al servizio 3D Secure
String[8]	**Brand** Brand carta di credito es. (VISA, MASTERCARD,…)

4.1.3 Classe IgfsCgSelector

La classe **`IgfsCgSelector`** viene utilizzata per ottenere la lista dei possibili strumenti di pagamento legati al selettore richiesto.

Sommario Properties Input		
Tipo[Dimensione]	**Dato**	**Proprietà**
URL	Obbligatorio	**ServerURL** Indirizzo del server di destinazione della richiesta
Integer	Opzionale	**Timeout** Timeout massimo espresso in millisecondi di completamento di una richiesta
String[64]	Obbligatorio	**KSig** Chiave per firmare il messaggio
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**ShopID** Chiave esterna identificante il pagamento
String[256]	Opzionale	**ShopUserRef** Identificativo chiente
PURCHASE, AUTH, VERIFY	Obbligatorio	**TrType** Tipologia di una richiesta AUTH = Pre Autorizzazione PURCHASE = Autorizzazione a livello contabile VERIFY = Verifica stato carta (nessun addebito o blocco plafond)
Long[12]	Condizionale	**Amount** Importo in virgola virtuale (es. 100 = 1,00 EUR) Obbligatorio con **TrType** PURCHASE o AUTH Opzionale con **TrType** VERIFY
EUR	Condizionale	**CurrencyCode** Valuta Obbligatorio con **TrType** PURCHASE o AUTH Opzionale con **TrType** VERIFY

IT, EN	Obbligatorio	**LangID** Lingua relativa alla pagina di inserimento dei dati sensibili associata ad una richiesta
String[256]	Opzionale	**AddInfo1** Campo a disposizione dell' Esercente
String[256]	Opzionale	**AddInfo2** Campo a disposizione dell' Esercente
String[256]	Opzionale	**AddInfo3** Campo a disposizione dell' Esercente
String[256]	Opzionale	**AddInfo4** Campo a disposizione dell' Esercente
String[256]	Opzionale	**AddInfo5** Campo a disposizione dell' Esercente

Metodi	
Tipo[Dimensione]	**Proprietà**
Boolean	**execute**() Esegue la transazione
	resetFields() Azzeramento parametri di richiesta

Sommario Properties Output	
Tipo[Dimensione]	**Proprietà**
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione di un errore/anomalia
TerminalInfo[]	**Terminal** Terminali abilitati al pagamento (0..*)

TerminalInfo	
Tipo[Dimensione]	**Proprietà**
String[16]	**Tid** Codice terminale dell'Esercente
String[256]	**Description** Descrizione codice terminale dell'Esercente
String[2]	**PayInstr** Codice della modalità di pagamento
String[64]	**PayInstrDescription** Descrizione della modalità di pagamento (Label del bottone)
URL[512][]	**ImgUrl** Url associato all'immagine (0..*)

4.2 Servizi diretti

4.2.1 Classe IgfsCgConfirm

La classe **`IgfsCgConfirm`** viene utilizzata per movimentare una autorizzazione effettata.

Sommario Properties Input		
Tipo[Dimensione]	**Dato**	**Proprietà**
URL	Obbligatorio	**ServerURL** Indirizzo del server di destinazione della richiesta
Integer	Opzionale	**Timeout** Timeout massimo espresso in millisecondi di completamento di una richiesta
String[64]	Obbligatorio	**KSig** Chiave per firmare il messaggio
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**ShopID** Chiave esterna identificante il pagamento
Long[12]	Obbligatorio	**Amount** Importo in virgola virtuale (es. 100 = 1,00 EUR)
Long[16]	Obbligatorio	**RefTranID** Codice Ordine relativo alla transazione da movimentare
Boolean	Opzionale	**SplitTran** è true se la conferma è parziale

Metodi	
Tipo[Dimensione]	**Proprietà**
Boolean	**execute**() Esegue la transazione
	resetFields() Azzeramento parametri di richiesta

Sommario Properties Output	
Tipo[Dimensione]	**Proprietà**
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione dell'errore/anomalia
Long[16]	**TranID** Codice Ordine processato
String[256]	**AddInfo1** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo2** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo3** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo4** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo5** Dati inviati in fase di autorizzazione dall'Esercente
Long[16]	**PendingAmount** Eventuale importo non confermato

4.2.2 Classe IgfsCgVoidAuth

La classe **`IgfsCgVoidAuth`** viene utilizzata per stornare una operazione di autorizzazione.

Sommario Properties Input		
Tipo[Dimensione]	**Dato**	**Proprietà**
URL	Obbligatorio	**ServerURL** Indirizzo del server di destinazione della richiesta
Integer	Opzionale	**Timeout** Timeout massimo espresso in millisecondi di completamento di una richiesta
String[64]	Obbligatorio	**KSig** Chiave per firmare il messaggio
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**ShopID** Chiave esterna identificante il pagamento
Long[12]	Obbligatorio	**Amount** Importo in virgola virtuale (es. 100 = 1,00 EUR)
Long[16]	Obbligatorio	**RefTranID** Codice Ordine relativo alla transazione da annullare

Metodi	
Tipo[Dimensione]	**Proprietà**
Boolean	**execute**() Esegue la transazione
	resetFields() Azzeramento parametri di richiesta

Sommario Properties Output	
Tipo[Dimensione]	**Proprietà**
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione dell'errore/anomalia
Long[16]	**TranID** Codice Ordine processato
String[256]	**AddInfo1** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo2** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo3** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo4** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo5** Dati inviati in fase di autorizzazione dall'Esercente

4.2.3 Classe IgfsCgCredit

La classe **`IgfsCgCredit`** viene utilizzata per riaccreditare una operazione di movimentazione.

Sommario Properties Input		
Tipo[Dimensione]	**Dato**	**Proprietà**
URL	Obbligatorio	**ServerURL** Indirizzo del server di destinazione della richiesta
Integer	Opzionale	**Timeout** Timeout massimo espresso in millisecondi di completamento di una richiesta
String[64]	Obbligatorio	**KSig** Chiave per firmare il messaggio
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**ShopID** Chiave esterna identificante il pagamento
Long[12]	Obbligatorio	**Amount** Importo in virgola virtuale (es. 100 = 1,00 EUR)
Long[16]	Obbligatorio	**RefTranID** Codice Ordine relativo alla transazione da riaccreditare
Boolean	Opzionale	**SplitTran** è true se la conferma è parziale

Metodi	
Tipo[Dimensione]	**Proprietà**
Boolean	**execute**() Esegue la transazione
	resetFields() Azzeramento parametri di richiesta

Sommario Properties Output	
Tipo[Dimensione]	**Proprietà**
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione dell'errore/anomalia
Long[16]	**TranID** Codice Ordine processato
String[256]	**AddInfo1** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo2** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo3** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo4** Dati inviati in fase di autorizzazione dall'Esercente
String[256]	**AddInfo5** Dati inviati in fase di autorizzazione dall'Esercente

4.3 Funzionalità batch

4.3.1 Classe IgfsBatchSubmit

La classe **`IgfsBatchSubmit`** viene utilizzata per inviare il file di conferma, ovvero le azione da intraprendere sulle transazioni specificate (conferma, storno, credito).

Sommario Properties Input		
Tipo[Dimensione]	**Dato**	**Proprietà**
URL	Obbligatorio	**ServerURL** Indirizzo del server di destinazione della richiesta
Integer	Opzionale	**Timeout** Timeout massimo espresso in millisecondi di completamento di una richiesta
String[64]	Obbligatorio	**KSig** Chiave per firmare il messaggio
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**BatchShopID** Codice Batch associato alla richiesta
File	Obbligatorio	**BatchDataFile** Path File conferme

Sommario Metodi	
Tipo[Dimensione]	**Proprietà**
Boolean	**execute**() Esegue la transazione
	resetFields() Azzeramento parametri di richiesta

Sommario Properties Output	
Tipo[Dimensione]	**Proprietà**
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione di un errore/anomalia
String[18]	**BatchID** Identificatore Batch

4.3.2 Classe IgfsBatchFetch

La classe **IgfsBatchFetch** viene utilizzata per prelevare il file di risposta al file conferme inviato tramite **IgfsBatchSubmit**.

Sommario Properties Input		
Tipo[Dimensione]	**Dato**	**Proprietà**
URL	Obbligatorio	**ServerURL** Indirizzo del server di destinazione della richiesta
Integer	Opzionale	**Timeout** Timeout massimo espresso in millisecondi di completamento di una richiesta
String[64]	Obbligatorio	**KSig** Chiave per firmare il messaggio
String[16]	Obbligatorio	**Tid** Codice terminale dell'Esercente
String[256]	Obbligatorio	**BatchShopID** Codice Batch associato alla richiesta dall'Esercente
File	Obbligatorio	**BatchDataFile** Path File esiti

Sommario Metodi	
Tipo[Dimensione]	**Proprietà**
Boolean	**execute**() Esegue la transazione
	resetFields() Azzeramento parametri di richiesta

Sommario Properties Output	
Tipo[Dimensione]	**Proprietà**
String[16]	**Rc** Esito della richiesta
String[80]	**ErrorDesc** Descrizione di un errore/anomalia
String[32]	**Status** Stato elaborazione Batch Possibili valori: *NOT_PROCESSED* *PROCESSING* *PROCESSED* *ERROR*
Long	**Size** Dimensione del file risposta

4.4 Esempi di implementazione con API

Al fine di facilitare la comprensione dei metodi relativi all'utilizzo delle API sopra descritte.

4.4.1 Pagamenti online

4.4.1.1 Java Init

```
// ====================================================================
// =          importazione classi di riferimento                      =
// ====================================================================
<%@page import="it.netsw.apps.igfs.cg.coms.api.init.IgfsCgInit" %>
<%@page import="it.netsw.apps.igfs.cg.coms.api.init.IgfsCgInit.*" %>
<%@page import="java.net.URL" %>
<%@page import="java.security.SecureRandom" %>
<%@page import="java.util.HashMap" %>
<%@page import="java.util.Map" %>
<%@page import="java.util.Properties" %>
<%@page import="java.io.InputStream" %>

<%
// ====================================================================
// = impostazione parametri per l'inizializzazione richiesta di       =
// = pagamento.                                                       =
// = NB: I parametri riportati sono solo a titolo di esempio          =
// ====================================================================
String serverURL     = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
int timeout          = 15000;

String tid           = "UNI_ECOM"; //per servizio MyBank usare UNI_MYBK
                                   //per AccountToPay utilizzare UNI_UNIPAY
                                   //per BancomatPat utilizzare UNI_BPAY

String kSig          = "UNI_TESTKEY";
String shopID        = "5687010820272485455"; // Chiave esterna UNIVOCA
                        identificante il pagamento
String email         = "user@customer.it";
TrType trType        = TrType.AUTH;
CurrencyCode curCode = CurrencyCode.EUR;
LangID langID        = LangID.IT;
long amount          = 100;
String errorURL      = "https://Esercente/error.jsp";
String notifyURL     = "https://Esercente/notify.jsp";

IgfsCgInit init = new IgfsCgInit();
init.setServerURL(new URL(serverURL));
init.setTimeout(timeout);

init.setTid(tid);
init.setKSig(kSig);
init.setShopID(shopID);
init.setShopUserRef(email);
```

```
init.setTrType(trType);
init.setCurrencyCode(curCode);
init.setLangID(langID);
init.setAmount(amount);
init.setErrorURL(new URL(errorUrl));
init.setNotifyURL(new URL(notifyURL));

// ===================================================================
// =               esecuzione richiesta di inizializzazione          =
// ===================================================================
if (!init.execute()) {
     // ============================================================================
     // = redirect del client su pagina di errore definita dall'Esercente  =
     // ============================================================================
     response.sendRedirect(errorURL + "?rc=" + init.getRc() + "&errorDesc=" +
                           init.getErrorDesc());
    return;
}

String paymentID = init.getPaymentID();
// NOTA: Salvo il paymentID relativo alla richiesta (es. sul DB)...

// ===================================================================
// =               redirect del client verso URL PagOnline BuyNow          =
// ===================================================================
URL redirectURL = init.getRedirectURL();
response.sendRedirect(redirectURL.toString());
%>
```

4.4.1.2 Java Verify

```
<%
// ========================================================================
// =          importazione classi di riferimento                          =
// ========================================================================
%>
<%@page import="it.netsw.apps.igfs.cg.coms.api.init.IgfsCgVerify" %>
<%@page import="java.net.URL" %>
<%@page import="java.security.SecureRandom" %>
<%@page import="java.util.HashMap" %>
<%@page import="java.util.Map" %>
<%@page import="java.util.Properties" %>
<%@page import="java.io.InputStream" %>
<%

// ========================================================================
// = impostazione parametri per l'inizializzazione richiesta di           =
// = pagamento.                                                           =
// = NB: I parametri riportati sono solo a titolo di esempio              =
// ========================================================================
String serverURL     = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
int timeout          = 15000;

String tid           = "UNI_ECOM"; //per servizio MyBank usare UNI_MYBK
String kSig          = "UNI_TESTKEY";
String shopID        = "5687010820272485455"; // Chiave esterna UNIVOCA
                       identificante il pagamento
String paymentID     = // NOTA: Leggo il paymentID rilasciato in fase di init (es.
                       dal DB)...
String errorURL      = "https://Esercente/error.jsp";
String esitoURL      = "https://Esercente/esito.jsp";

IgfsCgVerify verify = new IgfsCgVerify();
verify.setServerURL(new URL(serverURL));
verify.setTimeout(timeout);
verify.setTid(tid);
verify.setKSig(kSig);
verify.setShopID(shopID);
verify.setPaymentID(paymentID);



// ========================================================================
// =              esecuzione richiesta di verifica                        =
// ========================================================================
if (!verify.execute()) {
  // ======================================================================
  // = redirect del client su pagina di errore definita dall'Esercente  =
  // ======================================================================
  response.sendRedirect(errorURL + "?rc=" + verify.getRc() + "&errorDesc=" +
                        verify.getErrorDesc());
  return;
}
```

```
// ========================================================================
// =          redirect del client verso URL Esito Pagamento Esercente  =
// ========================================================================
StringBuffer resultUrl = new StringBuffer();
resultUrl.append(esitoURL);
resultUrl.append("?rc=" + verify.getRc());
resultUrl.append("&tranID=" + verify.getTranID());
resultUrl.append("&enrStatus=" + verify.getEnrStatus());
resultUrl.append("&authStatus=" + verify.getAuthStatus());
response.sendRedirect(resultUrl.toString());
%>
```

4.4.1.3 .NET Init

```
// ====================================================================
// =          importazione classi di riferimento                      =
// ====================================================================
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Text;
using it.netsw.apps.igfs.cg.coms.api.init;

// ====================================================================
// = impostazione parametri per l'inizializzazione richiesta di       =
// = pagamento.                                                       =
// = NB: I parametri riportati sono solo a titolo di esempio          =
// ====================================================================
String serverURL     = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
int timeout          = 15000;

String tid           = "UNI_ECOM"; //per servizio MyBank usare UNI_MYBK
String kSig          = "UNI_TESTKEY";
String shopID        = "56870108202724855455"; // Chiave esterna UNIVOCA
                       identificante il pagamento
String email         = "user@customer.it";
TrType trType        = TrType.AUTH;
CurrencyCode curCode = CurrencyCode.EUR;
LangID langID        = LangID.IT;
long amount          = 100;
String errorURL      = "https://Esercente/error.aspx";
String notifyURL     = "https://Esercente/notify.ashx";

IgfsCgInit init = new IgfsCgInit();
init.ServerURL = new Uri(serverURL);
init.Timeout = timeout;
init.Tid = tid;
init.KSig = kSig;
init.ShopID = shopID;
init.ShopUserRef = email;
init.TrType = trType;
init.CurrencyCode = curCode;
init.LangID = langID;
init.Amount = amount;
init.ErrorURL = new Uri(errorUrl);
init.NotifyURL = new Uri(notifyURL);
```

```
// ====================================================================
// =              esecuzione richiesta di inizializzazione            =
// ====================================================================
if (!init.execute()) {
  // ================================================================
  // = redirect del client su pagina di errore definita dall'Esercente  =
  // ================================================================
  Response.Redirect(errorURL + "?rc=" + init.Rc + "&errorDesc=" + init.ErrorDesc);
  return;
}

String paymentID = init.PaymentID;
// NOTA: Salvo il paymentID relativo alla richiesta (es. sul DB)...

// ====================================================================
// =          redirect del client verso URL PagOnline Imprese         =
// ====================================================================
Uri redirectURL = init.RedirectURL;
Response.Redirect(redirectURL.ToString());
```

4.4.1.4 .NET Verify

```
// ====================================================================
// =           importazione classi di riferimento                     =
// ====================================================================
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Text;
using it.netsw.apps.igfs.cg.coms.api.init;

// ====================================================================
// = impostazione parametri per l'inizializzazione richiesta di       =
// = pagamento.                                                       =
// = NB: I parametri riportati sono solo a titolo di esempio          =
// ====================================================================
String serverURL     = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
int timeout          = 15000;

String tid           = "UNI_ECOM"; //per servizio MyBank usare UNI_MYBK
String kSig          = "UNI_TESTKEY";
String shopID        = "5687010820272485455"; // Chiave esterna UNIVOCA
                        identificante il pagamento
String paymentID     = // NOTA: Leggo il paymentID rilasciato in fase di init (es.
                          dal DB)...
String errorURL      = "https://Esercente/error.aspx";
String esitoURL      = "https://Esercente/esito.aspx";

IgfsCgVerify verify = new IgfsCgVerify();
verify.ServerURL = new Uri(serverURL);
verify.Timeout = timeout;
verify.Tid = tid;
verify.KSig = kSig;
verify.ShopID = shopID;
verify.PaymentID = paymentID;

// ====================================================================
// =              esecuzione richiesta di verifica                    =
// ====================================================================
if (!verify.execute()) {
  // ================================================================
  // = redirect del client su pagina di errore definita dall'Esercente  =
  // ================================================================
  Response.Redirect(errorURL + "?rc=" + verify.Rc + "&errorDesc=" +
                    verify.ErrorDesc);
  return;
}
```

```
// ====================================================================
// =          redirect del client verso URL Esito Pagamento Esercente  =
// ====================================================================
StringBuilder resultUrl = new StringBuilder();
resultUrl.Append(esitoURL);
resultUrl.Append("?rc=" + verify.Rc);
resultUrl.Append("&tranID=" + verify.TranID);
resultUrl.Append("&enrStatus=" + verify.EnrStatus);
resultUrl.Append("&authStatus=" + verify.AuthStatus);
Response.Redirect(resultUrl.ToString());
```

4.4.1.5 PHP Init

```php
<?php
// ===================================================================
// =          importazione classi di riferimento                     =
// ===================================================================
require('IGFS_CG_API/init/IgfsCgInit.php');

// ===================================================================
// = impostazione parametri per l'inizializzazione richiesta di       =
// = pagamento.                                                       =
// = NB: I parametri riportati sono solo a titolo di esempio          =
// ===================================================================
$init = new IgfsCgInit();
$init->serverURL = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
$init->timeout = 15000;

$init->tid = "UNI_ECOM"; //per servizio MyBank usare UNI_MYBK
$init->kSig = "UNI_TESTKEY";
$init->shopID = "5687010820272485455"; // Chiave esterna UNIVOCA identificante il
pagamento
$init->shopUserRef = "user@customer.it";
$init->trType = "AUTH";
$init->currencyCode = "EUR";
$init->amount = 100;
$init->langID = "IT";
$init->notifyURL = "https://Esercente/notify.php";
$init->errorURL = "https://Esercente/error.php";

// ===================================================================
// =              esecuzione richiesta di inizializzazione            =
// ===================================================================
if (!$init->execute()) {
  // ================================================================
  // = redirect del client su pagina di errore definita dall'Esercente  =
  // ================================================================
  header("location: error.php?rc=".urlencode($init->rc) . "&errorDesc=" .
         urlencode($init->errorDesc));
  return;
}

// NOTA: Salvo il $init->paymentID relativo alla richiesta (es. sul DB)...

// ===================================================================
// =              redirect del client verso URL PagOnline BuyNow          =
// ===================================================================
header("location: ".$init->redirectURL);
?>
```

4.4.1.6 PHP Verify

```php
<?php
// ========================================================================
// =          importazione classi di riferimento                          =
// ========================================================================
require('IGFS_CG_API/init/IgfsCgVerify.php');

// ========================================================================
// = impostazione parametri per l'inizializzazione richiesta di           =
// = pagamento.                                                           =
// = NB: I parametri riportati sono solo a titolo di esempio              =
// ========================================================================
$verify = new IgfsCgVerify();
$verify->serverURL = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
$verify->timeout = 15000;

$verify->tid = "UNI_ECOM"; //per servizio MyBank usare UNI_MYBK
$verify->kSig = "UNI_TESTKEY";
$verify->shopID = "5687010820272485455"; // Chiave esterna UNIVOCA identificante il
                  pagamento
$verify->paymentID = // NOTA: Leggo il paymentID rilasciato in fase di init (es. dal
                   DB)...
$errorURL = "https://Esercente/error.php";
$esitoURL = "https://Esercente/esito.php";

// ========================================================================
// =              esecuzione richiesta di verifica                        =
// ========================================================================
if (!$verify->execute()) {
  // ======================================================================
  // = redirect del client su pagina di errore definita dall'Esercente    =
  // ======================================================================
  header("location: ".$errorURL . "?rc=" . $verify->rc . "&errorDesc=" . $verify-
         >errorDesc);
  return;
}

// ========================================================================
// =          redirect del client verso URL Esito Pagamento Esercente     =
// ========================================================================
header("location: ".$esitoURL . "?esito=OK&rc=" . $verify->rc . "&tranID=" .
       $verify->tranID . "&enrStatus=" . $verify->enrStatus . "&authStatus=" .
       $verify->authStatus);
?>
```

4.4.2 Pagamenti diretti per carte di credito

4.4.2.1 Java Confirm

```
// ======================================================================
// =          importazione classi di riferimento                        =
// ======================================================================
<%@page import="it.netsw.apps.igfs.cg.coms.api.tran.BaseIgfsCgTran" %>
<%@page import="it.netsw.apps.igfs.cg.coms.api.tran.IgfsCgConfirm" %>
<%@page import="it.netsw.apps.igfs.cg.coms.api.tran.IgfsCgConfirm.*" %>
<%@page import="java.net.URL" %>
<%@page import="java.io.InputStream" %>
<%
// ======================================================================
// = impostazione parametri per l'inizializzazione richiesta di         =
// = confirm.                                                           =
// = NB: I parametri riportati sono solo a titolo di esempio            =
// ======================================================================
String serverURL     = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
int timeout          = 15000;

String tid           = "UNI_ECOM";
String kSig          = "UNI_TESTKEY";
String shopID        = "5687010820272485455"; // Chiave esterna UNIVOCA
                        identificante il pagamento
String errorURL      = "https://Esercente/error.jsp";
long refTranID       = // Identificativo transazione autorizzata
long amount          = 100;

IgfsCgConfirm confirm = new IgfsCgConfirm();
confirm.setServerURL(new URL(serverURL));
confirm.setTimeout(timeout);
confirm.setTid(tid);
confirm.setKSig(kSig);
confirm.setShopID(shopID);
confirm.setRefTranID(refTranID);
confirm.setAmount(amount);

// ======================================================================
// =              esecuzione richiesta                                  =
// ======================================================================
if (!confirm.execute()) {
  // ====================================================================
  // = redirect del client su pagina di errore definita dall'Esercente  =
  // ====================================================================
    response.sendRedirect(errorURL + "?rc=" + confirm.getRc() + "&errorDesc=" +
                          confirm.getErrorDesc());
    return;
}
// ======================================================================
// = presa in carico esito transazione per poi proseguire               =
// ======================================================================
String result = confirm.getRc();
%>
```

4.4.2.2 Java Void

```
// ====================================================================
// =          importazione classi di riferimento                      =
// ====================================================================
<%@page import="it.netsw.apps.igfs.cg.coms.api.tran.BaseIgfsCgTran" %>
<%@page import="it.netsw.apps.igfs.cg.coms.api.tran.IgfsCgVoidAuth" %>
<%@page import="it.netsw.apps.igfs.cg.coms.api.tran.IgfsCgVoidAuth.*" %>
<%@page import="java.net.URL" %>
<%@page import="java.io.InputStream" %>
<%
// ====================================================================
// = impostazione parametri per l'inizializzazione richiesta di       =
// = void.                                                          =
// = NB: I parametri riportati sono solo a titolo di esempio          =
// ====================================================================
String serverURL     = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
int timeout          = 15000;

String tid           = "UNI_ECOM";
String kSig          = "UNI_TESTKEY";
String shopID        = "56870108202724854555"; // Chiave esterna UNIVOCA
                        identificante il pagamento
String errorURL      = "https://Esercente/error.jsp";
long refTranID       = // Identificativo transazione autorizzata
long amount          = 100;

IgfsCgVoidAuth voidAuth = new IgfsCgVoidAuth();
voidAuth.setServerURL(new URL(serverURL));
voidAuth.setTimeout(timeout);
voidAuth.setTid(tid);
voidAuth.setKSig(kSig);
voidAuth.setShopID(shopID);
voidAuth.setRefTranID(refTranID);
voidAuth.setAmount(amount);

// ====================================================================
// =              esecuzione richiesta                                =
// ====================================================================
if (!voidAuth.execute()) {
  // ================================================================
  // = redirect del client su pagina di errore definita dall'Esercente  =
  // ================================================================
    response.sendRedirect(errorURL + "?rc=" + voidAuth.getRc() + "&errorDesc=" +
                          voidAuth.getErrorDesc());
    return;
}
// ====================================================================
// = presa in carico esito transazione per poi proseguire             =
// ====================================================================
String result = voidAuth.getRc();
%>
```

4.4.2.3 Java Credit

```
// ========================================================================
// =          importazione classi di riferimento                          =
// ========================================================================
<%@page import="it.netsw.apps.igfs.cg.coms.api.tran.BaseIgfsCgTran" %>
<%@page import="it.netsw.apps.igfs.cg.coms.api.tran.IgfsCgCredit" %>
<%@page import="it.netsw.apps.igfs.cg.coms.api.tran.IgfsCgCredit.*" %>
<%@page import="java.net.URL" %>
<%@page import="java.io.InputStream" %>
<%
// ========================================================================
// = impostazione parametri per l'inizializzazione richiesta di           =
// = credito.                                                             =
// = NB: I parametri riportati sono solo a titolo di esempio              =
// ========================================================================
String serverURL      = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
int timeout           = 15000;

String tid            = "UNI_ECOM";
String kSig           = "UNI_TESTKEY";
String shopID         = "56870108202724854555"; // Chiave esterna UNIVOCA
                          identificante il pagamento
String errorURL       = "https://Esercente/error.jsp";
long refTranID        = // Identificativo transazione autorizzata
long amount           = 100;

IgfsCgCredit credit = new IgfsCgCredit();
credit.setServerURL(new URL(serverURL));
credit.setTimeout(timeout);
credit.setTid(tid);
credit.setKSig(kSig);
credit.setShopID(shopID);
credit.setRefTranID(refTranID);
credit.setAmount(amount);

// ========================================================================
// =              esecuzione richiesta                                    =
// ========================================================================
if (!credit.execute()) {
  // ======================================================================
  // = redirect del client su pagina di errore definita dall'Esercente    =
  // ======================================================================
    response.sendRedirect(errorURL + "?rc=" + credit.getRc() + "&errorDesc=" +
                          credit.getErrorDesc());
    return;
}
// ========================================================================
// = presa in carico esito credito per poi proseguire                     =
// ========================================================================
String result = credit.getRc();
%>
```

4.4.2.4 .NET Confirm

```
// ====================================================================
// =          importazione classi di riferimento                      =
// ====================================================================
using System;
using System.Collections;
using System.Data;
using System.Web;
using System.Text;
using it.netsw.apps.igfs.cg.coms.api.tran;
// ====================================================================
// = impostazione parametri per l'inizializzazione richiesta di       =
// = confirm.                                                         =
// = NB: I parametri riportati sono solo a titolo di esempio          =
// ====================================================================
String serverURL     = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
int timeout          = 15000;

String tid           = "UNI_ECOM";
String kSig          = "UNI_TESTKEY";
String shopID        = "5687010820272485455"; // Chiave esterna UNIVOCA identificante
                       il pagamento
String errorURL      = "https://Esercente/error.aspx";
long refTranID       = // Identificativo transazione autorizzata
long amount          = 100;

IgfsCgConfirm confirm = new IgfsCgConfirm();
confirm.ServerURL = new Uri (serverURL);
confirm.Timeout   = timeout;
confirm.Tid       = tid;
confirm.KSig      = kSig;
confirm.ShopID    = shopID;
confirm.RefTranID = refTranID;
confirm.Amount    = amount;

// ====================================================================
// =              esecuzione richiesta                                =
// ====================================================================
if (!confirm.execute()) {
  // ================================================================
  // = redirect del client su pagina di errore definita dall'Esercente  =
  // ================================================================
    Server.Transfer(errorURL + "?rc=" + confirm.Rc + "&errorDesc=" +
                    confirm.ErrorDesc);
    return;
}
// ====================================================================
// = presa in carico esito transazione per poi proseguire             =
// ====================================================================
String result = confirm.Rc;
```

4.4.2.5 .NET Void

```
// ====================================================================
// =          importazione classi di riferimento                      =
// ====================================================================
using System;
using System.Collections;
using System.Data;
using System.Web;
using System.Text;
using it.netsw.apps.igfs.cg.coms.api.tran;
// ====================================================================
// = impostazione parametri per l'inizializzazione richiesta di       =
// = void     .                                                       =
// = NB: I parametri riportati sono solo a titolo di esempio          =
// ====================================================================
String serverURL     = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
int timeout          = 15000;

String tid           = "UNI_ECOM";
String kSig          = "UNI_TESTKEY";
String shopID        = "5687010820272485455"; // Chiave esterna UNIVOCA
                        identificante il pagamento
String errorURL      = "https://Esercente/error.aspx";
long refTranID       = // Identificativo transazione autorizzata
long amount          = 100;

IgfsCgVoidAuth voidAuth = new IgfsCgVoidAuth();
voidAuth.ServerURL = new Uri (serverURL);
voidAuth.Timeout   = timeout;
voidAuth.Tid       = tid;
voidAuth.KSig      = kSig;
voidAuth.ShopID    = shopID;
voidAuth.RefTranID = refTranID;
voidAuth.Amount    = amount;

// ====================================================================
// =              esecuzione richiesta                                =
// ====================================================================
if (!voidAuth.execute()) {
  // ================================================================
  // = redirect del client su pagina di errore definita dall'Esercente  =
  // ================================================================
    Server.Transfer(errorURL + "?rc=" + voidAuth.Rc + "&errorDesc=" +
                    voidAuth.ErrorDesc);
    return;
}
// ====================================================================
// =        presa in carico esito void per poi proseguire             =
// ====================================================================
String result = voidAuth.Rc;
```

4.4.2.6 .NET Credit

```
// ====================================================================
// =          importazione classi di riferimento                      =
// ====================================================================
using System;
using System.Collections;
using System.Data;
using System.Web;
using System.Text;
using it.netsw.apps.igfs.cg.coms.api.tran;
// ====================================================================
// = impostazione parametri per l'inizializzazione richiesta di       =
// = credito.                                                         =
// = NB: I parametri riportati sono solo a titolo di esempio          =
// ====================================================================
String serverURL      = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
int timeout           = 15000;

String tid            = "UNI_ECOM";
String kSig           = "UNI_TESTKEY";
String shopID         = "5687010820272485455"; // Chiave esterna UNIVOCA
                          identificante il pagamento
String errorURL       = "https://Esercente/error.aspx";
long refTranID        = // Identificativo transazione autorizzata
long amount           = 100;

IgfsCgCredit credit = new IgfsCgCredit();
credit.ServerURL    = new Uri (serverURL);
credit.Timeout      = timeout;
credit.Tid          = tid;
credit.KSig         = kSig;
credit.ShopID       = shopID;
credit.RefTranID    = refTranID;
credit.Amount       = amount;

// ====================================================================
// =              esecuzione richiesta                                =
// ====================================================================
if (!credit.execute()) {
  // ================================================================
  // = redirect del client su pagina di errore definita dall'Esercente  =
  // ================================================================
    Server.Transfer(errorURL + "?rc=" + credit.Rc + "&errorDesc=" +
                    credit.ErrorDesc);
    return;
}
// ====================================================================
// = presa in carico esito credit per poi proseguire                  =
// ====================================================================
String result = credit.Rc;
```

4.4.3 Funzionalità batch

4.4.3.1 Java Submit

```
// ========================================================================
// =          importazione classi di riferimento                          =
// ========================================================================
import java.io.File;
import java.net.URL;
import it.netsw.apps.igfs.cg.coms.api.batch.*;

// ========================================================================
// = impostazione parametri per la sottomissione del file                 =
// = NB: I parametri riportati sono solo a titolo di esempio              =
// ========================================================================
String serverURL = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
int timeout = 15000;

String tid = "UNI_ECOM";
String kSig = "UNI_TESTKEY";
String batchShopID = "BATCH_1";
File batchDataFile = new File("../files/file.mov");
String batchID = null;

IgfsCgBatchSubmit submit = new IgfsCgBatchSubmit();
submit.setServerURL(new URL(serverURL));
submit.setTimeout(timeout);
submit.setTid(tid);
submit.setKSig(kSig);
submit.setBatchShopID(batchShopID);
submit.setBatchDataFile(batchDataFile);

// ========================================================================
// =              esecuzione richiesta di sottomissione                   =
// ========================================================================
if (!submit.execute()) {
      System.out.println("Submit execute error");
       System.out.println("rc=" + submit.getRc() + " errorDesc=" +
                          submit.getErrorDesc());
} else {
  batchID = submit.getBatchID();
}
```

4.4.3.2 Java Fetch

```
// ========================================================================
// =          importazione classi di riferimento                          =
// ========================================================================
import java.net.URL;
import it.netsw.apps.igfs.cg.coms.api.batch.*;
import it.netsw.apps.igfs.cg.coms.api.batch.IgfsCgBatchFetch.Status;

// ========================================================================
// = impostazione parametri per la sottomissione del file                 =
// = NB: I parametri riportati sono solo a titolo di esempio              =
// ========================================================================
String serverURL = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";
int timeout = 15000;

String tid = "UNI_ECOM";
String kSig = "UNI_TESTKEY";
String batchShopID = "BATCH_1";
File batchDataFile = new File("../files/file.res");

IgfsCgBatchFetch fetch = new IgfsCgBatchFetch();
fetch.setServerURL(new URL(serverURL));
fetch.setTimeout(timeout);
fetch.setTid(tid);
fetch.setKSig(kSig);
fetch.setBatchShopID(batchShopID);
fetch.setBatchDataFile(batchDataFile);

// ========================================================================
// =     esecuzione richiesta di acquisizione file esiti                  =
// ========================================================================
if (!fetch.execute()) {
  System.out.println("Fetch execute error. rc= "+ fetch.getRc()+ " -  "+
                     fetch.getErrorDesc());
}

if (fetch.getStatus() == Status.PROCESSED) {
  // Read batchDataFile...
}
```

4.4.3.3 .NET Submit

```
// ====================================================================
// =          importazione classi di riferimento                      =
// ====================================================================
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Text;
using it.netsw.apps.igfs.cg.coms.api.init;

// ====================================================================
// = impostazione parametri per la sottomissione del file             =
// = NB: I parametri riportati sono solo a titolo di esempio          =
// ====================================================================
String serverURL = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";

String tid = "UNI_ECOM";
String kSig = "UNI_TESTKEY";
String batchShopID = "batchID1";
String batchDataFile = "c:\data.mov";
String batchID = null;

IgfsCgBatchSubmit submit = new IgfsCgBatchSubmit();
submit.ServerURL = new Uri(serverURL);
submit.Tid = tid;
submit.KSig = kSig;
submit.BatchShopID = batchShopID;
submit.BatchDataFile = batchDataFile;

// ====================================================================
// =              esecuzione richiesta di sottomissione               =
// ====================================================================
if (!submit.execute())
{
  System.console.WriteLine("Error rc=" + submit.Rc + " errorDesc=" +
                           submit.ErrorDesc);
}
else
{
  batchID = submit.BatchID;
}
```

4.4.3.4 .NET Fetch

```
// ====================================================================
// =          importazione classi di riferimento                      =
// ====================================================================
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Text;
using it.netsw.apps.igfs.cg.coms.api.batch;

// ====================================================================
// = impostazione parametri per la sottomissione del file             =
// = NB: I parametri riportati sono solo a titolo di esempio          =
// ====================================================================
String serverURL = "https://testeps.netswgroup.it/UNI_CG_SERVICES/services";

String tid = "UNI_ECOM";
String kSig = "UNI_TESTKEY";
String batchShopID = "batchID1";
String batchDataFile = "c:\data.res";

IgfsCgBatchFetch fetch = new IgfsCgBatchFetch();
fetch.ServerURL = new Uri(serverURL);
fetch.Tid = tid;
fetch.KSig = kSig;
fetch.BatchShopID = batchShopID;
fetch.BatchDataFile = batchDataFile;

// ====================================================================
// =     esecuzione richiesta di acquisizione file esiti              =
// ====================================================================
if (!fetch.execute())
{
  System.console.WriteLine("Error rc=" + fetch.Rc + " errorDesc=" +
                           fetch.ErrorDesc);
}

if (fetch.Status == Status.PROCESSED)
{
  // Read batchDataFile...
}
```

APPENDICE

Appendice A: calcolo signature

L' HMACSHA256 è un tipo di algoritmo con chiave costruito dalla funzione hash SHA-256 e utilizzato come codice HMAC (Hash-based Message Authentication Code).

A titolo di esempio, riportiamo il calcolo dell'algoritmo HMACSHA256 sul campo "signature" attraverso la tecnologia java:

```
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;

….

// ================================================================
//   calcolo della signature attraverso l'algoritmo SHA256
//
//   key    = chiave segreta
//   fields = parametri del messaggio
// ================================================================
public String getSHA256Signature(String key, Object... fields) throws Exception {
       StringBuilder sb = new StringBuilder();
       for (Object field : fields) {
              if (field != null) {
                     sb.append(field.toString());
              }
       }

       byte data[] = sb.toString().getBytes();

       String alg = "HmacSHA256";
       SecretKeySpec sk1 = new SecretKeySpec(key.getBytes(), alg);
       Mac mac = Mac.getInstance(alg);
       mac.init(sk1);
       byte sig[] = mac.doFinal(data);
       return new String(Base64.encode(sig));
}
```

e attraverso la tecnologia .NET

```
 using System.Security.Cryptography;
...

        static String getSignature(String key, params Object[] fields)
        {
            StringBuilder sb = new StringBuilder();
            for (int i = 0; i < fields.Length; i++)
            {
                Object field = fields[i];
                if (field != null)
                {
                    sb.Append(field.ToString());
                }
            }
            byte[] keyByte = Encoding.UTF8.GetBytes(key);
            HMACSHA256 hmacsha256 = new HMACSHA256(keyByte);
            byte[] data = Encoding.UTF8.GetBytes(sb.ToString());
            byte[] mac = hmacsha256.ComputeHash(data);
            return System.Convert.ToBase64String(mac);
        }
```

NB: La classe `Base64` implementa l'encoding in Base64 dei byte componenti la signature.

Appendice B: File Batch

Di seguito viene illustrata la struttura dei files batch per operare in **PagOnline Imprese.**

Ogni transazione è rappresentata da un record del file, i campi sono separati dalla virgola “,”.

Nel caso in cui un campo sia opzionale e non inserito sul file deve essere inserito il campo vuoto.

La struttura inviata in input presenta già i campi vuoti che verranno valorizzati in fase di output.

FILE DI INPUT

La struttura di INPUT segue lo schema:

TRTYPE,TID,AMOUNT,CURRENCY,REFTRANID,SHOPID, , , ,
,SHOPUSERREF,ADDINFO1,ADDINFO2,ADDINFO3,ADDINFO4,ADDINFO5,TOPUPID

CAMPO	DESCRIZIONE	M/O	NOTE
TRTYPE	Tipo di operazione da effettuare	M	**D**=DEBIT (CONFERMA) **V**=VOID (STORNO) **C**=CREDIT (CREDITO) **A**=AUTH (AUTORIZZAZIONE) **P**=PURCHASE (ADDEBITO)
TID	Codice terminale	M	Codice terminale utilizzato per la transazione.
AMOUNT	Importo della transazione con il carattere “punto” (.) come separatore dei decimali	M	ES: 1 Euro => “1.00”. Si utilizza il numero di decimali previsti dagli standard per la relativa valuta.
CURRENCY	Codice numerico identificativo della valuta	M	ES: 978 per Euro.
REFTRANID	Riferimento TranID	O	Per TRTYPE “D” o “V” il REFTRANID atteso è quello ottenuto in fase di autorizzazione attraverso la proprietà TRANID (Nel caso di TRTYPE “D” relativo ad operazioni di TOPUP tale valore non deve essere impostato). Per TRTYPE “C” deve essere inserito il campo TRANID ottentuo sulla conferma contabile. Per TRTYPE “A” o “P” non deve essere impostato.
SHOPID	Chiave esterna identificante l’operazione	M	
SHOPUSERREF	Identificativo cliente	O	ES: email.
ADDINFO1	Campo a disposizione dell‘Esercente	O	
ADDINFO2	Campo a disposizione dell‘Esercente	O	
ADDINFO3	Campo a disposizione dell‘Esercente	O	

CAMPO	DESCRIZIONE	M/O	NOTE
ADDINFO4	Campo a disposizione dell'Esercente	O	
ADDINFO5	Campo a disposizione dell'Esercente	O	
TOPUPID	ID riferimento TopUp	O	Per TRTYPE "A" può essere impostato per identificare un operazione di TOPUP. Per TRTYPE "D" deve essere impostato col valore utilizzato per le transazioni di autorizzazione.

FILE DI OUTPUT

La struttura di OUTPUT segue lo schema:

```
TRTYPE,TID,AMOUNT,CURRENCY,REFTRANID,SHOPID, ,
RC,DESC,TRANID,SHOPUSERREF,ADDINFO1,ADDINFO2,ADDINFO3,ADDINFO4,ADDINFO5,TOPUPID,
```

CAMPO	DESCRIZIONE	M/O	NOTE
TRTYPE	Tipo di operazione da effettuare	M	REPLICATO DA FILE DI INPUT
TID	Codice terminale	M	REPLICATO DA FILE DI INPUT
AMOUNT	Importo della transazione con il carattere “punto” (.) come separatore dei decimali	M	REPLICATO DA FILE DI INPUT
CURRENCY	Codice numerico identificativo della valuta	M	REPLICATO DA FILE DI INPUT
REFTRANID	Riferimento TranID	O	REPLICATO DA FILE DI INPUT
SHOPID	Chiave esterna identificante l’operazione	M	REPLICATO DA FILE DI INPUT
RC	Codice di risposta	M	Vedi APPENDICE C: CODICI RITORNO.
DESC	Descrizione RC	M	Descrizione del codice di risposta, in caso di errore fornisce informazioni aggiuntive.
TRANID	TranID	M	Fornisce un riferimento alla transazione di effettuata. Il dato è presente solo in caso di azione conclusa correttamente.
SHOPUSERREF	Identificativo cliente	O	REPLICATO DA FILE DI INPUT
ADDINFO1	Campo a disposizione dell‘Esercente	O	REPLICATO DA FILE DI INPUT
ADDINFO2	Campo a disposizione dell‘Esercente	O	REPLICATO DA FILE DI INPUT
ADDINFO3	Campo a disposizione dell‘Esercente	O	REPLICATO DA FILE DI INPUT
ADDINFO4	Campo a disposizione dell‘Esercente	O	REPLICATO DA FILE DI INPUT
ADDINFO5	Campo a disposizione dell‘Esercente	O	REPLICATO DA FILE DI INPUT
TOPUPID	ID riferimento TopUp	O	REPLICATO DA FILE DI INPUT

Appendice C: Codici Ritorno

CODICE RITORNO STATO POSITIVO

CODICE	DESCRIZIONE
IGFS_000	TRANSAZIONE OK

CODICE RITORNO STATO PENDENTE

CODICE	DESCRIZIONE
IGFS_890	TRANSAZIONE IN CORSO (stato operazione indefinito)
IGFS_814	TRANSAZIONE IN CORSO

CODICE RITORNO STATO NEGATIVO

CODICE	DESCRIZIONE
IGFS_001	DESTINATARIO SCONOSCIUTO
IGFS_00155	ID BATCH NON VALIDO
IGFS_00156	ID BATCH NON UNIVOCO
IGFS_00157	STRUMENTO PAGAMENTO NON VALIDO
IGFS_00158	NUMERO CARTA NON NUMERICO
IGFS_00159	NUMERO CARTA NON PRESENTE
IGFS_002	CARTA SCADUTA
IGFS_00260	L'IMPORTO DEL CREDITO SUPERA L'IMPORTO DEL MOVIMENTO
IGFS_00261	L'IMPORTO DEL MOVIMENTO SUPERA L'IMPORTO DELL' AUTORIZZAZIONE
IGFS_003	CARTA ERRATA
IGFS_004	CARTA IN BLACK LIST
IGFS_00452	CODICE TERMINALE NON PRESENTE
IGFS_00456	CODICE TERMINALE ERRATO
IGFS_005	ERRORE DI FORMATO
IGFS_006	ERRORE FILE SYSTEM
IGFS_007	ERRORE DI COMUNICAZIONE
IGFS_00701	BATCH ID NON PROCESSATO
IGFS_00704	BATCH ID NON NUMERICO
IGFS_00705	BATCH ID NON PRESENTE
IGFS_008	AUTORIZZAZIONE NEGATA
IGFS_009	RITIRARE CARTA
IGFS_00950	DIRECTORY BATCH UPLOAD NON PRESENTE
IGFS_00951	DIRECTORY BATCH DOWNLOAD NON PRESENTE
IGFS_00952	NOME DIRECTORY ARCHIVIAZIONE BATCH NON PRESENTE
IGFS_010	ESERCENTE NON ABILITATO

CODICE	DESCRIZIONE
IGFS_01000	TRANSAZIONE NEGATA DAL SISTEMA ANTIFRODE
IGFS_011	CONTATTARE ACQUIRER
IGFS_014	ESERCENTE NON CONVENZIONATO
IGFS_015	CARTA NON GESTITA
IGFS_016	CARTA IN RANGE NEGATIVO O STRANIERA
IGFS_018	CARTA INESISTENTE
IGFS_020	CARTA INVALIDA
IGFS_021	CODICE ESERCENTE ERRATO
IGFS_029	DATA SCADENZA ERRATA
IGFS_030	FONDI INSUFFICIENTI
IGFS_032	IMPORTO NON VALIDO
IGFS_033	TRANSAZIONE ORIGINALE NON TROVATA
IGFS_083	ERRORE CIFRATURA TRANSAZIONE
IGFS_085	CODICE DIVISA ERRATO
IGFS_086	MALFUNZIONAMENTO SISTEMA
IGFS_087	ACQUIRER NON RAGGIUNGIBILE
IGFS_088	MANCATA RISPOSTA DA ACQUIRER
IGFS_091	MALFUNZIONAMENTO SISTEMA ACQUIRER
IGFS_092	TRANSAZIONE SCONOSCIUTA
IGFS_093	CONFERMA GIA' PRESENTE
IGFS_095	STORNO PER NOTIFICA INESISTENTE
IGFS_096	STORNO PER AUTORIZZAZIONE INESISTENTE
IGFS_097	CONFERMA PER AUTORIZZAZIONE INESISTENTE
IGFS_098	IMPORTO SUPERIORE AD IMPORTO AUTORIZZATO
IGFS_10000	CARATTERI NON VALIDI
IGFS_101	MAC ERRATO
IGFS_102	SOSPETTA FRODE
IGFS_104	CARTA SOGGETTA A RESTRIZIONI
IGFS_107	CONTATTARE ISSUER
IGFS_108	CONTATTARE ISSUER: CASO SPECIALE
IGFS_112	INSERIRE PIN
IGFS_115	FUNZIONE NON SUPPORTATA SU CARTA
IGFS_117	PIN ERRATO
IGFS_118	CONTO NON TROVATO O NON ABILITATO
IGFS_119	OPERAZIONE NON PERMESSA AL TITOLARE
IGFS_121	SUPERATO LIMITE IMPORTO
IGFS_122	ERRORE SICUREZZA
IGFS_123	SUPERATO LIMITE FREQUENZA
IGFS_125	CARTA NON ATTIVA
IGFS_129	SOSPETTA FRODE SU CARTA

CODICE	DESCRIZIONE
IGFS_160	CARTA PERSA
IGFS_164	DATA ANTEC. A BLOCCO CARTA
IGFS_180	DATI ERRATI
IGFS_181	DATI SENSIBILI ERRATI
IGFS_1921	3DS: IMPOSSIBILE AUTENTICARE CARTA (PARES=U)
IGFS_1922	3DS: AUTENTICAZIONE NON AVVENUTA (PARES=N)
IGFS_1923	3DS: IMPOSSIBILE VERIFICARE ISCRIZIONE CARTA (VERES=U)
IGFS_20000	DATI MANCANTI
IGFS_20001	CODICE OPERAZIONE NON VALIDO
IGFS_20002	SESSIONE SCADUTA
IGFS_20007	STATO ORDINE NON VALIDO
IGFS_20010	URL INVIO RISPOSTA NON VALIDO
IGFS_20011	URL INVIO ERRORE NON VALIDO
IGFS_20012	SHOPID NON VALIDO
IGFS_20013	CODICE LINGUA NON VALIDO
IGFS_20014	CAMPO AGGIUNTIVO NON VALIDO
IGFS_20018	CVV2 NON VALIDO
IGFS_20019	SHOPID NON VALIDO
IGFS_20020	CAMPO ADDIZIONALE NON VALIDO
IGFS_20021	CAMPO API VERSION NON VALIDO
IGFS_20022	CAMPO SIGNATURE NON VALIDO
IGFS_20023	CAMPO PAYMENT ID NON VALIDO
IGFS_20024	CODICE AUTORIZZAZIONE MANCANTE
IGFS_20025	CAMPO REFERENCE DATA NON VALIDO
IGFS_20026	SHOP ID DUPLICATO
IGFS_20027	RICHIESTA BATCH NON VALIDA
IGFS_20028	DATI BATCH MANCANTI
IGFS_20029	DATI BATCH NON VALIDI
IGFS_20030	DIRECTORY DATI BATCH NON VALIDA
IGFS_20031	DATI BATCH DUPLICATI
IGFS_20032	NOME BATCH FILE NON VALIDO
IGFS_20033	DATI BATCH NON TROVATI
IGFS_20034	BATCH SHOPID NON VALIDO
IGFS_20035	ID ORDINE NON VALIDO
IGFS_20036	PAN NON VALIDO
IGFS_20037	CVV2 NON VALIDO
IGFS_20038	DATA SCADENZA ERRATA
IGFS_20044	DESCRIZIONE PAGAMENTO NON VALIDA
IGFS_20090	TRANSAZIONE CANCELLATA DALL'UTENTE
IGFS_20092	NUMERO DI TELEFONO NON REGISTRATO

CODICE	DESCRIZIONE
IGFS_20100	ERRORE NOTIFICA ESERCENTE
IGFS_400	STORNO OK
IGFS_800	TERMINALE NON ABILITATO
IGFS_801	BANCA SELEZIONATA ERRATA
IGFS_802	TENTATIVI PIN ESAURITI
IGFS_803	CODICE TERMINALE ERRATO
IGFS_804	CHIAVE DISALLINEATA
IGFS_805	ERRORE CIFRATURA
IGFS_807	TERMINALE CHIUSO
IGFS_808	TERMINALE NON CHIUSO
IGFS_809	ERRORE SEQUENZA
IGFS_810	TERMINALE NON RICONOSCIUTO
IGFS_811	TERMINALE BLOCCATO
IGFS_812	TERMINALE CHIUSO FORZATAMENTE
IGFS_813	OPERAZIONE NON PERMESSA
IGFS_815	CARTA BLOCCATA
IGFS_90000	DATABASE ERROR
IGFS_90005	TIMESTAMP ERRATO
IGFS_902	TRANSAZIONE NON VALIDA
IGFS_903	REINVIARE TRANSAZIONE
IGFS_907	EMITTENTE NON ADERENTE
IGFS_908	DESTINAZIONE NON TROVATA
IGFS_909	ERRORE DI SISTEMA
IGFS_910	SISTEMA ISSUER NON ATTIVO
IGFS_911	TIME OUT
IGFS_912	ISSUER NON RAGGIUNGIBILE
IGFS_913	TRANSAZIONE DUPLICATA
IGFS_990	STRUMENTO PAGAMENTO NON ATTIVO

Appendice D: Messaggi d’esempio di Popolazione Dati

PaymentInitGateway Init Message

REQUEST

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ser="http://services.api.web.cg.igfs.apps.netsw.it/">
<soapenv:Body>
<ser:Init>
<request>
      <tid>TEST_SELECTOR</tid>
      <signature>ky9FZ06pRvjTyLexcK5TEfK/Vm5R6WvxVJuSOANGTVs=</signature>
      <shopID>1701061569</shopID>
      <shopUserRef>cliente@email.it</shopUserRef>
      <trType>AUTH</trType>
      <amount>100</amount>
      <currencyCode>EUR</currencyCode>
      <langID>IT</langID>
      <notifyURL>http://merchant/notify.jsp</notifyURL>
      <errorURL>http://merchant/show.jsp</errorURL>
      <level3Info>
            <destinationName>Nome Destinatario</destinationName>
            <destinationStreet>Indirizzo</destinationStreet>
            <destinationCity>Citta</destinationCity>
            <destinationState>Provincia</destinationState>
            <destinationPostalCode>20100</destinationPostalCode>
            <destinationCountryCode>ITA</destinationCountryCode>
            <freightAmount>10</freightAmount>
            <taxAmount>10</taxAmount>
            <product>
                  <productCode>ProductCode1</productCode>
                  <productDescription>ProductDescription1</productDescription>
                  <items>2</items>
                  <amount>10</amount>
            </product>
            <product>
                  <productCode>ProductCode2</productCode>
                  <productDescription>ProductDescription2</productDescription>
                  <items>2</items>
                  <amount>10</amount>
            </product>
            <product>
                  <productCode>ProductCode3</productCode>
                  <productDescription>ProductDescription3</productDescription>
                  <items>2</items>
                  <amount>10</amount>
            </product>
            <product>
                  <productCode>ProductCode4</productCode>
                  <productDescription>ProductDescription4</productDescription>
                  <items>2</items>
                  <amount>10</amount>
            </product>
      </level3Info>
      <description>Descrizione per transazione -1701061569</description>
</request>
</ser:Init>
</soapenv:Body>
</soapenv:Envelope>
```

RESPONSE

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Body>
<ns1:InitResponse xmlns:ns1="http://services.api.web.cg.igfs.apps.netsw.it/">
<response xmlns:ns2="http://services.api.web.cg.igfs.apps.netsw.it/">
      <tid>TEST SELECTOR</tid>
      <rc>IGFS_000</rc>
      <error>false</error>
      <errorDesc></errorDesc>
      <signature>sORxvFHSWoq/IQEematagsaULZ0tbdviIXQbJNnkChQ=</signature>
      <shopID>1701061569</shopID>
      <paymentID>344517213104948134</paymentID>
      <redirectURL>https://IPGATEWAY/IPS_CG_WEB/app/main/show?referenceData=5CA44411481B07D1</redirectURL>
</response>
</ns1:InitResponse>
</soap:Body>
</soap:Envelope>
```

PaymentInitGateway Verify Message

REQUEST

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ser="http://services.api.web.cg.igfs.apps.netsw.it/">
<soapenv:Body>
<ser:Verify>
<request>
      <tid>TEST_SELECTOR</tid>
      <signature>9KCe+jA4z8AElX2TJUipxTE5bk7Q7DOX79I1qDnNQkc=</signature>
      <shopID>1701061569</shopID>
      <paymentID>344517213104948134</paymentID>
</request>
</ser:Verify>
</soapenv:Body>
</soapenv:Envelope>
```

RESPONSE

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Body>
<ns1:VerifyResponse xmlns:ns1="http://services.api.web.cg.igfs.apps.netsw.it/">
<response xmlns:ns2="http://services.api.web.cg.igfs.apps.netsw.it/">
      <tid>TEST_ECOM</tid>
      <payInstr>CC</payInstr>
      <rc>IGFS_000</rc>
      <error>false</error>
      <errorDesc>TRANSAZIONE OK</errorDesc>
      <signature>Ois3yADPsyWTuHf/zgBc4WQd7ZvziIDavp2K1XLpBkI=</signature>
      <shopID>1701061569</shopID>
      <paymentID>344517213104948134</paymentID>
      <tranID>3066511440512679</tranID>
      <authCode>24973 </authCode>
      <enrStatus>Y</enrStatus>
      <authStatus>Y</authStatus>
```

RESPONSE

```
      <brand>VISA</brand>
      <maskedPan>401200******1112</maskedPan>
      <payInstrToken>548051F6AE29013134AA3DF3E7CE6FC5</payInstrToken>
      <expireMonth>XX</expireMonth>
      <expireYear>XXXX</expireYear>
      <level3Info>
            <destinationName>Nome Destinatario</destinationName>
            <destinationStreet>Indirizzo</destinationStreet>
            <destinationCity>Citta</destinationCity>
            <destinationState>Provincia</destinationState>
            <destinationPostalCode>20100</destinationPostalCode>
            <destinationCountryCode>ITA</destinationCountryCode>
            <freightAmount>10</freightAmount>
            <taxAmount>10</taxAmount>
            <product>
                  <productCode>ProductCode1</productCode>
                  <productDescription>ProductDescription1</productDescription>
                  <items>2</items>
                  <amount>10</amount>
            </product>
            <product>
                  <productCode>ProductCode2</productCode>
                  <productDescription>ProductDescription2</productDescription>
                  <items>2</items>
                  <amount>10</amount>
            </product>
            <product>
                  <productCode>ProductCode3</productCode>
                  <productDescription>ProductDescription3</productDescription>
                  <items>2</items>
                  <amount>10</amount>
            </product>
            <product>
                  <productCode>ProductCode4</productCode>
                  <productDescription>ProductDescription4</productDescription>
                  <items>2</items>
                  <amount>10</amount>
            </product>
      </level3Info>
</response>
</ns1:VerifyResponse>
</soap:Body>
</soap:Envelope>
```