BetSoft Games Integration Interface

Common Wallet

Version 3.07

Updated 2015-05-12

INTEGRATION DESCRIPTION

Introduction

This document describes the Common Wallet model of integration between BetSoft Gaming and a client system. The following chapters describe the general concept of integration as well as descriptions and examples of the API methods used for the Common Wallet integration.

Terms and Abbreviations

The following terms and abbreviations are used in this document:

- **BSG** - BetSoftGaming.
- **BSG system, BSG server -** BSG game software/system, developed and supported by BSG, provider.
- **EC -** External Client, company/person who integrates BSG games.
- **EC system , EC server, ES -** software/system developed and supported by External Client.
- **CW –** Common Wallet
- **FRB –** Free Rounds Bonus
- **CM –** Casino Manager
- **COPY –** The Test or “staging” servers for the BSG system.
- **PJ –** Progressive Jackpot

General Notes

The BSG Games standard integration interface assumes the following:

a) BSG-EC systems are integrated in a B2B configuration, where each side has its own system/back-end.

b) The BSG-EC Common Wallet integration treats the EC system as the primary system, and the BSG system is the secondary system. This means the EC system manages the primary user accounts database, balance, personal information, payment operations etc. The BSG system manages only the data necessary to perform game operations. This data is either provided on game startup (login, registration), or requested by the BSG system from the EC system. Similarly, the BSG system manages the player balance within the bounds of the turn, requesting confirmation on each bet from the EC system and notifying it of wins.

c) Communication between the BSG-EC systems is performed through the Internet, using HTTP/HTTPS.

SubCasinos and Banks

Each EC is configured as separate system within the BSG COPY/LIVE cluster. In the BSG CM, these systems are called a “SubCasino”. For each SubCasino, BSG generates and provides its own domain that will be used in start game URLs and other URLs that can be called by the ES from the BSG server. The BSG server determines the SubCasino by the domain the request originates from. The BSG system can configure multiple aliases for each SubCasino. The EC can also provide their own domain to be configured as an alias for a SubCasino.

Each SubCasino has at least one Bank. Most configuration parameters are set on the Bank level: EC API configuration, game limits, coin denominations, Jackpots etc. An EC can have multiple sites / sub-systems associated with it. For each site / sub-system the BSG system can configure its own Bank. Banks are created by BSG at the request of the EC. BSG defines a numeric ID for each bank, but the EC can define its own ID for each bank. However, the EC must inform BSG about what ID should be used for each bank.

BSG Games

BSG provides many different games to be integrated into the EC site. BSG games (game clients) are available for several different platforms.

Desktop BSG game clients are implemented as Flash applets.

For Android ©, iOS ©, Windows Mobile© devices - BSG game clients are implemented using HTML5.

Each game has its own ID. BSG uses numeric game IDs (integer). By default the same game has a unique ID for each platform. For example:

- **210 – MrVegas (Flash, desktop)**
- **269 – MrVegas Mobile (iOS)**
- **270 – MrVegas Android**

NOTE: The EC may require that their own game IDs be used. In this case the EC must provide an ID for each BSG game. BSG uses string representation to store external game IDs so they can be numeric or text. E.g. “Game ID = 1234” or “Game ID = MrVegas”

Localization

BSG game clients are localized for many languages. Required localization can be passed as a“lang” parameter to all BSG start game URLs. Each bank has a default language defined. If a “lang” parameter is not passed in the start game URL, the game will startin the specified default language.

BSG can configure two possible behaviors for cases where a non-existent language was passed in the “lang” parameter in the start game URL:

- Show an error message that the specified language is not available.(This is the default option)
- Open the game using the default language, as specified by the Bank's parameters, instead of the language parameter passed in the start game URL.

Game List

BSG provides a special URL to obtain a list of all available games for a bank. Here is an example URL:

http://EC1DOMAIN/gamelist.do?bankId=EC1

When called, this URL returns an XML response with all the information about all games configured for the bank specified. It includes the game ID, name and supported languages.

XML example:

```
<GAMESSUITES>
<SUITES>
<SUITEID="Video Poker" NAME="Video Poker">
<GAMES>
<GAMENAME="Bonus Poker" ID="10137"
IMAGEURL="" LANGUAGES="en">
</GAME>
</GAMES>
</SUITE>
</SUITES>
</GAMESSUITES>
```

Integrating PC Games

BSG game clients for Desktop PCs (both Mac and Windows)are Flash applets loaded by a player's web browser. The method depends on the EC's front-end organization. For example, the following methods are generally used:

a) The EC web site provides a link to a BSG game startup page. The Player loads this startup page from the BSG web site. The page loads the Flash applet inside.

b) The Player loads the EC's Flash shell from the EC's web site. The Flash shell loads the Flash applet inside.

c) The Player loads the game's startup page from the EC's web site. The page loads the Flash applet inside.

d) The EC web site provides a link to the BSG game lobby page. The Player navigates to the BSG game lobby page, and then loads the game from the lobby.

In all cases, the EC system must provide all necessary startup parameters in the game startup URL.

The default option is "**a**". It does not require any additional implementation.

Integrating Mobile Games

Integrating Mobile Games has only one major restriction: Mobile Games cannot be opened in an iframe/div/span container. They should only be opened in their own window.

For more details about integrating mobile games, please refer to the "BSG Mobile Games Integration Guide" available on the BetSoft Gaming CPMA.

Starting Games

BSG provides two launch game URLs:

- **Launch game for a Guest**
  URL example:
  /cwguestlogin.do?gameId=231&lang=en&bankId=EC1

- **Launch game for an Authorized Player**
  URL example:
  /cwstartgamev2.do?token=12j2j2j3nsdnj3k&mode=real&gameId=191&lang=en&bankId=EC1

Launch URLs above are used for all BSG games (Desktop, iOS, Android).

Games can be started in FREE or REAL mode.

In FREE mode, the player is playing with Fun Money. The player's fun money balance is set to the default value upon starting a FREE mode game session. (The Initial Fun Money Balance can be configured at the EC's request. The default amount is 1000.00 of whatever the default currency for the originating bank is set to). FREE mode games are not logged or recorded on the BSG side. The player cannot continue an unfinished round for a FREE mode game. FREE mode games are played with the same configuration as for REAL mode, per the originating Bank's parameters. No calls to the ES are made during FREE mode games. FREE mode Progressive Jackpot games have separate Progressive Jackpot banks from any REAL mode PJ banks. FREE mode PJ banks are saved in server memory cache and are reset on server restart.

REAL mode games are played for Real money. BSG makes calls to the ES to inform it about all bet/wins made by the player. REAL mode games are logged and recorded on the BSG side. The player can continue an unfinished round next time they enter the game in REAL mode.

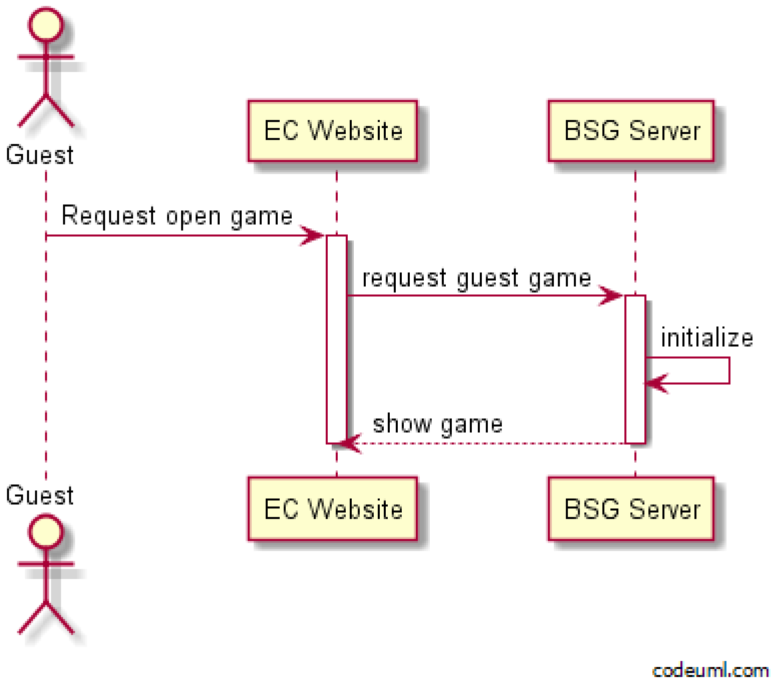
Games for Guests are always started in FREE mode. Authorized players can start games in REAL or FREE mode, at the EC's discretion.

Starting a Game for Guest Players

The Start Game URL for Guest Players has the following parameters:	
bankId	specifies from what bank the game was started. The configuration parameters of this bank will be used to start the game.
gameId	specifies what game should be started.
lang	specifies what localization should be used. (optional)
homeUrl	for mobile games only: specifies the URL where the player should be redirected on pressing HOME button. If this parameter is not passed, the default URL configured for the bank is used. (optional)

URL example:
http://EC1DOMAIN/cwguestlogin.do?gameId=231&lang=en&bankId=EC1

Starting a game for Guest mode is very simple. See diagram below.



Workflow:

- When the player presses the button/banner/link to launch the BSG game in guest mode on EC's Website, the EC Server can open the game differently depending on the however the prefers. For this example ,we will assume the link opens the game in a popup window.
- EC Website opens a new popup window using a BSG guest mode launch URL (see above).
- BSG initializes the game and shows a page with the embedded game client to the Player.

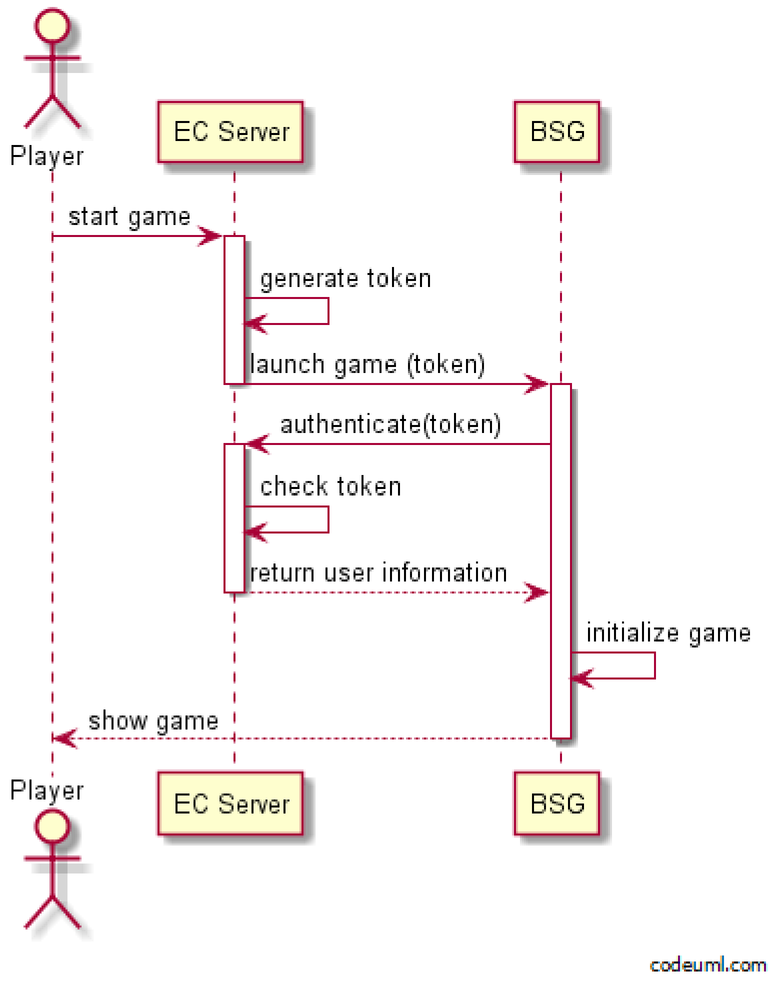
Starting a Game for Authorized Players

The Start Game URL for Authorized Players has following parameters:	
Token	used to verify that the player is authorized to play the game.
Bank Id	specifies from what bank the game was started. The configuration parameters of this bank will be used to start game.
Game Id	specifies what game should be started.
Mode	For an authorized player the game can be started in REAL or FREE mode.
Lang	specifies what localization should be used. (optional)
Home Url	for mobile games only: specifies the URL where the player should be redirected on pressing HOME button. If this parameter is not passed, the default URL configured for the bank is used. (optional)
Cashier Url	specifies the URL where the player should be redirected if they have no money in their balance and the "go to cashier" button is pressed. If this parameter is not passed, the default URL configured for the bank is used. (optional)

- URL Example:

/cwstartgamev2.do?token=12j2j2j3nlasdnj3k&mode=real&gameId=191&lang=en&bankId=EC1

BSG uses a token model to authenticate players. Here is diagram describing how starting game for an authorized player is performed:



Workflow:

- Player presses button/banner/link to launch a BSG game in REAL mode on the EC's Website. The EC Server can open the game differently (depending on how the EC prefers). For this example we will assume the game opens in a popup window.
- Before the EC server redirects player to the BSG Authorized launch URL it must generate a unique token and associate it with the player. See recommendations for generating token below.
- EC server opens a new window and redirects the Player to the BSG launch game URL passing the token as one of the parameters.
- BSG server receives the request and checks the token by making an Authenticate API request to the EC Server passing the received token as a parameter.
- EC Server authenticates the Player by token and returns information about the Player to the BSG Server.
- The BSG Server performs the login operation for the Player. In the event that a player starts a game for the first time and is not yet registered on the BSG side – The BSG server will make a getAccountInfo API call to get information about the player and then register them on the BSG side. After this the BSG server initializes the game for the Player and redirects the Player to the page with the embedded game client, as normal.

Recommendations for token:

- A new token should be generated each time player starts a game
- The token should automatically expire after a short period of time (several seconds) and at the moment player has left EC's site (session expired)
- The token should also expire as soon as it is used to authenticate the API request

Please note that the token is not hidden and can be "sniffed". So for security purposes the token must be at least: different for each session and only be valid for a limited time.

Mobile Detector

BSG offers a platform detection system to determine if a mobile device is being used to access a game. It works as follows:

- EC starts a Desktop game session. (provides gameId of Desktop game to launch game URL)
- BSG detects the platform that the user is playing from.
- Depending on platform detected, BSG starts the appropriate version of the game and launches the appropriate game client.

By default the mobile detector is enabled, but it can be disabled at the EC's request.

Game History

BSG provides a game history URL that can be used to show any game's history to player. By default the interface is not skinned, but it can be skinned at the EC's request.

URL example:

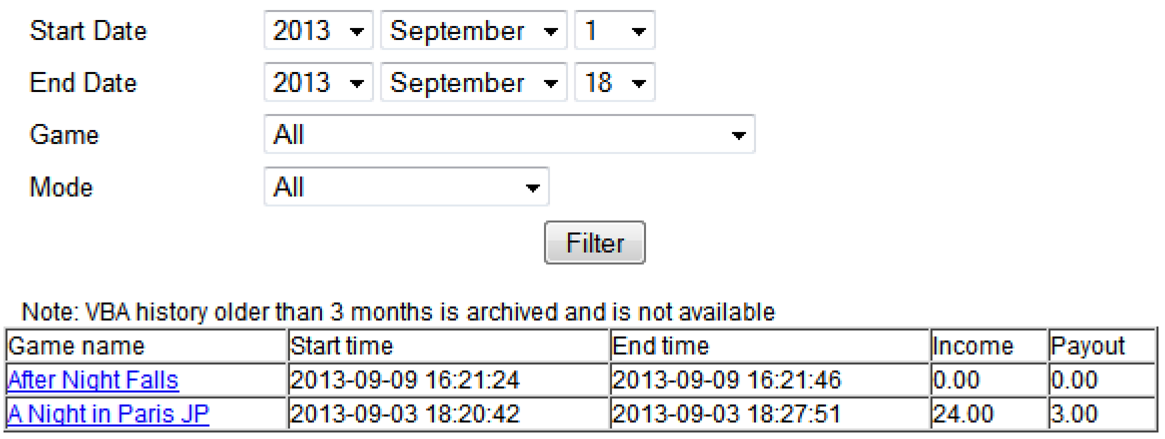
http://lobby.default.discreetgaming.com/cwstarthistory.do?bankId=271&token=jsdbsg21

Game history URL uses the same token authorization as start game URL.

Parameters:

- bankId
- token

Below is an example of how the default history is presented:



Start Date	2013	September	1
End Date	2013	September	18
Game	All		
Mode	All		

Note: VBA history older than 3 months is archived and is not available

Game name	Start time	End time	Income	Payout
After Night Falls	2013-09-09 16:21:24	2013-09-09 16:21:46	0.00	0.00
A Night in Paris JP	2013-09-03 18:20:42	2013-09-03 18:27:51	24.00	3.00

Winners Feed

BSG can be configured to collect information about big winners and provide this information to EC. It can be used to show a Winners feed on the EC's site. The Winners feed is provided only at the EC's request and is not configured by default.

Winners are collected by analyzing the results of a game session. The Player's win is calculated as follows: Total game session wins amount – Total game session bets amount.

It is possible to configure a minimum win threshold to be included into feed as well as the total number of records to show in Winners feed.

When a new player wins the same or a higher amount than the current lowest ranking winner that has been previously registered in the feed,the oldest record is removed and the newest entry is included.

The Winners feed is provided as xml that is updated by server once per minute.

XML example:

```
<WINNERS>
<GAMESESSION>
<NICKNAME>betsoftars</NICKNAME>
<GAMENAME>European Roulette</GAMENAME>
<GAMEREVENUE>3.00</GAMEREVENUE>
<TIME>2013-09-17 11:54:12</TIME>
<CURRENCY>ARS</CURRENCY>
</GAMESESSION>
</WINNERS>
```

Winners feed URL example:

http://lobby.default.discreetgaming.com/winners/winners_480.xml

Jackpot Feed

The Jackpot feed is an API that provides the current amounts of all available jackpots for all games in xml format. It can be used to build a section with information about jackpots on the EC's site. The Jackpot feed is configured at the EC's request, and is not configured by default.

The Jackpot feed can be configured to contain the total amount of all Jackpot banks for game+currency, or to contain information about each Jackpot Bank for each coin.

XML example with totals:

```
<jackpots>
<jackpotGame>
<gameId>231</gameId>
<gameName>Tycoons</gameName>
<currencyCode>EUR</currencyCode>
<jackpotAmount>145075.05</jackpotAmount>
</jackpotGame>
</jackpots>
```

XML example for each coin:

```
<jackpots>
<jackpotGame>
<gameId>269</gameId>
<gameName>Mr. Vegas Mobile</gameName>
<coin>0.02</coin>
<currencyCode>EUR</currencyCode>
<jackpotAmount>181.38</jackpotAmount>
</jackpotGame>
</jackpots>
```

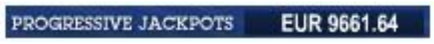
Jackpot feed URL example:

http://lobby.default.discreetgaming.com/jackpots/jackpots_480.xml

Jackpot Tickers

BSG provides Jackpot Tickers for each game implemented in Flash.

Example of ticker:



See example here:

http://lobby.default.discreetgaming.com/jackpotticker.jsp?bankId=394

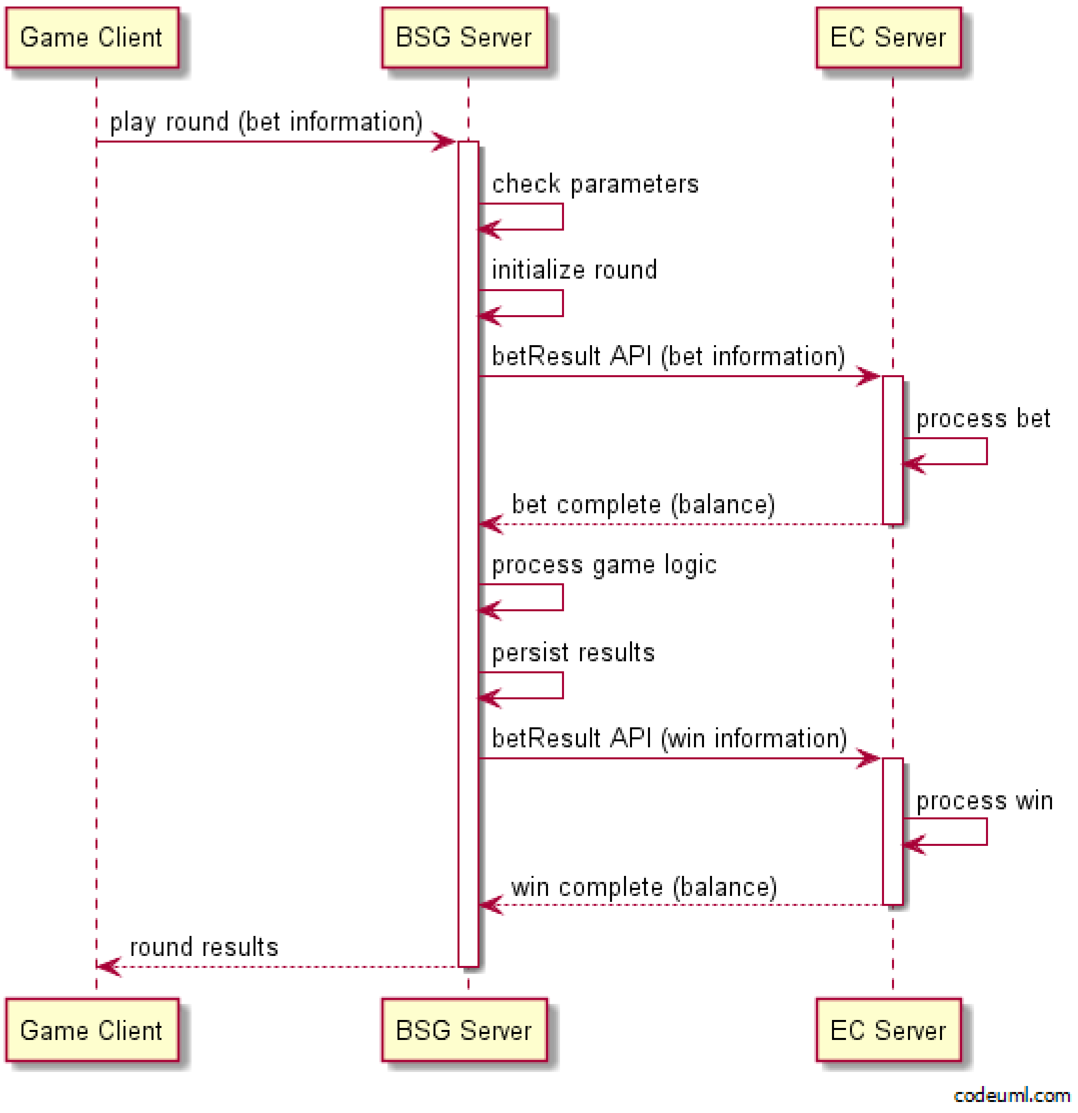
As an alternative option BSG can provide API to obtain amount of PJ for each game+currency. Information will be provided in following format: GBP 17414.36

REAL Mode Game Playing

A game can only be started in REAL mode for authorized players. The BSG server creates a *game session* for each real mode game. A Game Session is recorded as the time that the game client is opened until it has been closed by the player, or the session times out due to inactivity. REAL mode game sessions are logged and recorded. Each game session has a unique ID that is sent to the ES with each *Bet/Result* API call.

BSG uses the "*Bet/Result*" API to inform the ES about each wager the player has made and each win that the player receives during a REAL mode game session. Each Bet/Result API call contains information about one single wallet transaction (it can be either bet or win). Each wallet transaction has its own unique ID.

Here is simplified diagram that describes how the games are played:



Workflow of how actions are normally processed for a simple game (erroneous cases will be described below):
• Player selects bet amount and initiates round in the game client. Game client sends a request to BSG server specifying bet parameters.
• BSG server validates request parameters and initializes new round. Please note that BSG server does not validate balance at this stage. Balance must be checked by EC server during processing bet.
• BSG server creates wallet transaction for the bet and sends a Bet/Result API request to EC Server to get approval for a bet.
• EC server checks available balance, applies bet to player balance, persists bet information and respond with information about bet transaction status and new balance (with bet applied).
• BSG server updates wallet transaction and synchronizes player's balance.
• BSG server processes game logic (deal cards, spin reels etc), calculates payout and persists it on the BSG side. NOTE: Persisted rounds can't be rolled back. BSG supposes that the win can't be rejected by any reason.
• BSG server creates a wallet transaction for win and sends it using Bet/Result API to EC.
• EC server applies win to player balance, persists win in EC DB and responds with information about
• win transaction status and new balance (with win applied).
• BSG server updates wallet transaction and synchronizes player's balance.
• BSG server responds with round results to the game client.
• Game client animates round for player.

Rounds

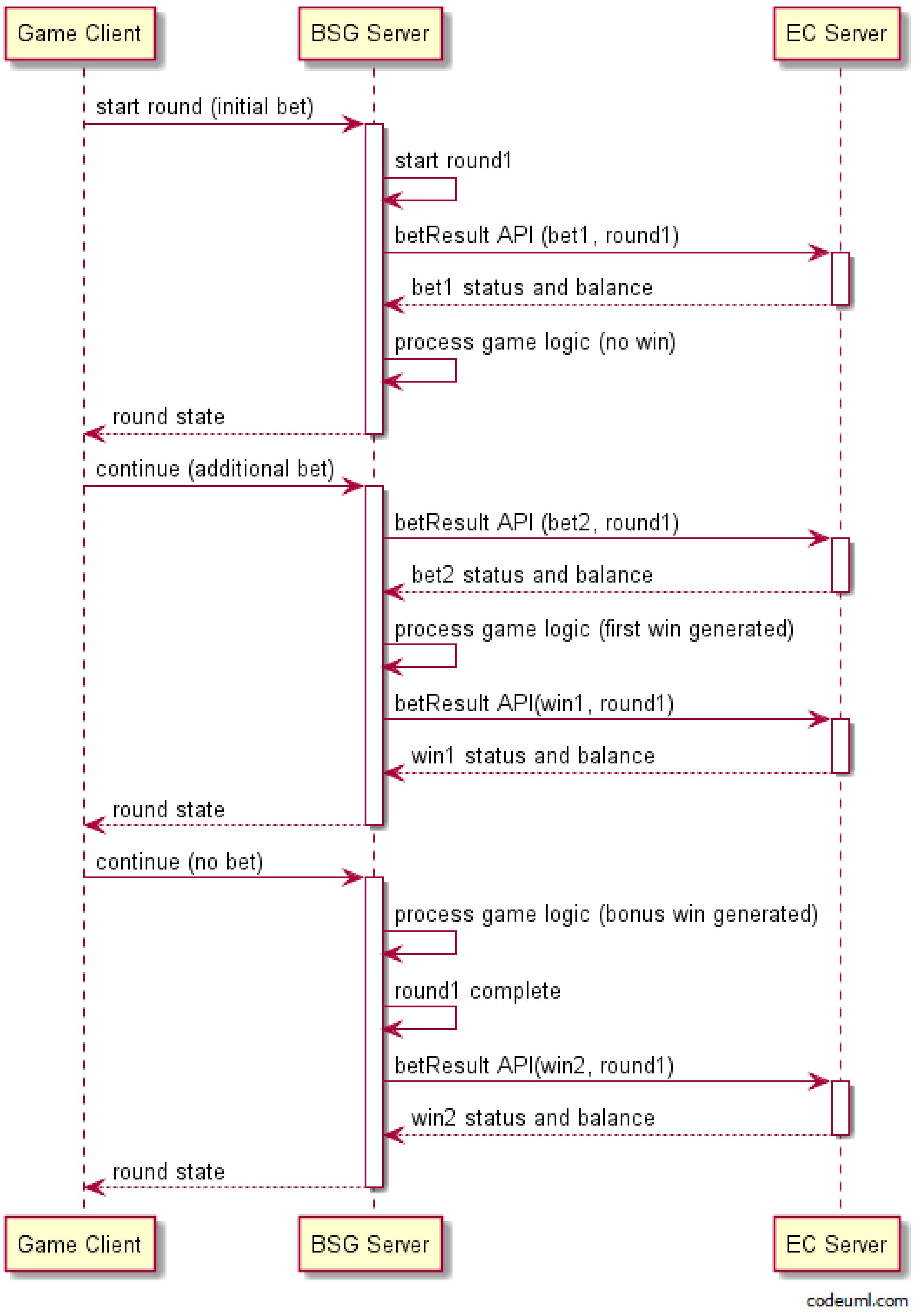
Above is a description of a very simple, single-step game. For most games, rounds require several actions from the player to be complete and can contain several bet/win transactions. The BSG server maintains rounds for each game and sends the ID of a round with each Bet/Result API call.

The BSG server persists the state of a round after each player action. It's possible that the player can leave the game with a pending (incomplete) round. In this case, the BSG server will allow the player to continue their round on next time they enter the same game. The round can be spread over several game sessions. Pending rounds can exist for several games at once (BSG does not force the player to complete a pending round in one game to play another game).

Some games allow the player to make additional bets within the same round. For example: Blackjack allows the player to double their bet with Double Up action. For such games, the BSG server can send several bet wallet transactions for the same round. Please note that the BSG server must check each bet the player makes – so the BSG server can't be configured to send only one bet for a round.

Rounds for some games can result in several wins. For example: the Click Me bonus in slot games. In this case, BSG can send several win wallet transactions for the same round by default. The BSG server can be configured to send only one single resulting win wallet transaction at the end of a round. We do not recommend using such configuration because wins will be not be applied on EC side, but will be applied on the BSG side in the game client. This can confuse the player.

Here is a diagram of a general round that can be played:


Game Client
BSG Server
EC Server
start round (initial bet)
start round1
betResult API (bet1, round1)
bet1 status and balance
process game logic (no win)
round state
continue (additional bet)
betResult API (bet2, round1)
bet2 status and balance
process game logic (first win generated)
betResult API(win1, round1)
win1 status and balance
round state
continue (no bet)
process game logic (bonus win generated)
round1 complete
betResult API(win2, round1)
win2 status and balance
round state
Game Client
BSG Server
EC Server
codeuml.com

In a general case, the BSG server works as following:
The player makes any action within the game client. The game client sends a request to the BSG server specifying what should be done (command with required parameters).
The BSG server loads the state of a round (if it exists) and checks the request parameters. In case a new round is started, BSG creates a new round.
The BSG server checks if the command received includes a bet. If it does, the BSG server creates wallet transaction for this bet and sendsa Bet/Result API request to the EC server.

The BSG server processes the game logic according to the command received and persists the round state/results.
In case the game logic resulted in any winnings, the BSG server creates a wallet transaction and sends it to ES.
The BSG server responds to the game client.

The BSG server can be configured to inform the EC server about the round that has ended. In this case, the BSG server always sends a Bet/Result API call with a win operation at the end of a round (even with zero amount) with roundFinished=true parameter.

Most games have rounds with one or more bet transactions at the beginning and zero or more win transactions at the end. But BSG has one exceptional game - Craps. Craps allows the player to put stakes on the table and leave them there for several dice drops (rounds). As a result - Craps can have rounds without any bets.

Negative Bets

Craps and Ride'm Poker games allow the player to remove a bet that was previously put on the table. BSG accounts such removing of bets as a decreasing of game session Income (not as a win) and calls it Negative Bets. The BSG server sends negative bets as separate parameter in Bet/Result API calls with a win transaction. Negative bets are not included in the win amount.

Processing of Bet/Result Requests on EC Server

In general, the EC server should process Bet/Result requests as follows:

- Check the request parameters and validate the hash.
- Check if the transaction was already processed. If yes – just return success response (See details below).
- Process operation: check balance, update player balance and save BSG wallet transaction if processing was successful.
- Return response to BSG.

First, that should make the EC server, when processing a Bet/Result request, check that the operation has not already been processed recently. For this check, the EC server must have complete BSG wallet transactions persisted for some safe time period (at least for 1 day). A BSG wallet transaction has a unique ID. The wallet transaction ID is sent with each Bet/Result request.

The EC server should check the operation by ID (other parameters should not be checked).In case the wallet transaction was already processed, the EC server must return a success response with the current player balance, and should not process the transaction again.

EC must check that player has enough of a balance when processing bet transactions. The BSG server does not check it. The BSG server has no actual player balance, only a “screenshot” of it from the last transaction. Balance can be reduced on the EC server while the player is playing a BSG game and the BSG server will not know about it. Please also note that BSG game clients do not allow for making bets higher than the known balance. In case the player has a low balance and made a deposit with an open BSG game client, their balance in the BSG game client will not be updated by default. The game needs to be restarted to update the balance.

Fail-safety

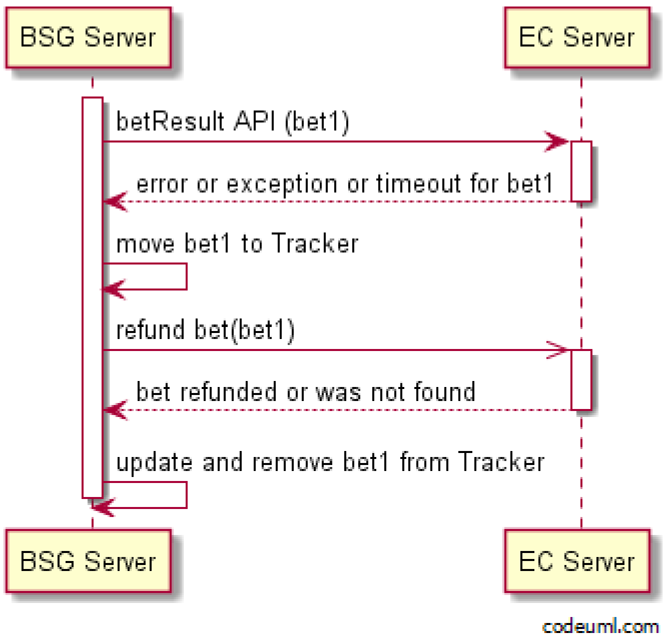
The BSG server always tries to resolve Bet/Result errors automatically. Resolving pending wallet transactions is completed by the Wallet Transaction Tracker. This Tracker is a multithreaded process, running on each cluster node, that periodically checks the status of pending transactions and tries to fix them.

NOTE: The player can’t play a BSG game while they have a pending transaction for it.

The wallet transaction is moved to the Tracker in case of ANY error (API error code in response, BSG application error, any other error such as network timeout, hardware error, etc).The Tracker works as follows: it has a list of pending transactions and periodically tries to resolve each transaction. In case the transaction was not resolved, it is left in the queue, and the Tracker will make another attempt in some time.

Bet and Win transactions are tracked differently.

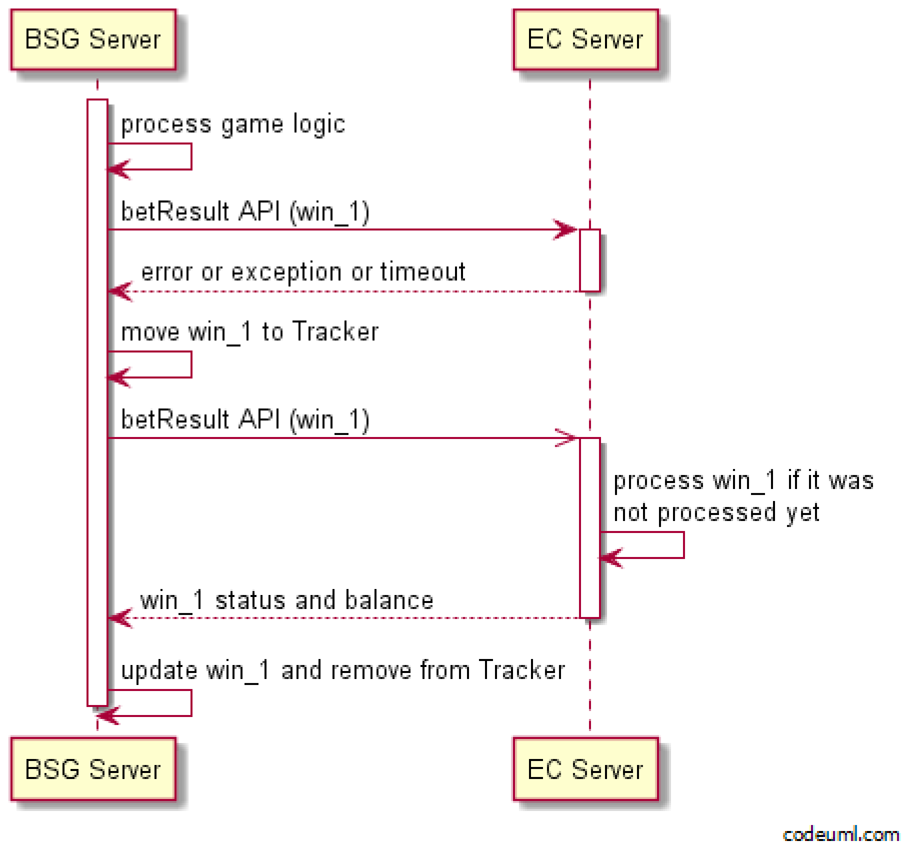
Bet tracking:



As can be seen from the gaming workflow, the bet transaction is performed before any game logic processing on the BSG side. So in case the bet is rejected by EC Server or any error occurred, the bet was not yet saved and processed on the BSG side. The BSG server stops any game logic execution in case it received any error on the bet and sends the bet to the Tracker.

The Tracker always tries to reject the bet if the bet exists and is complete on the EC server but does not try to complete it, if it is not yet completed. For bet tracking, BSG uses a special API method: refundBet. The BSG Tracker sends refundBet API requests until it gets a successful result: transaction not found error or successfully refunded.

Win tracking:



On the other hand, win transactions are created after the game logic was processed and the round state persisted. In case of error for win operation, the BSG server tries to complete it. The Tracker will send a betResult API call with all the same win transaction parameters as in the original request until the EC server successfully processes it.

The BSG tracker does not try to resolve transactions for an infinite amount of time. By default, the Tracker is configured to resolve transactions for a 48-hour period. In case the transaction was not resolved in 48 hours, the Tracker removes it from the tracking queue. The unresolved transaction is not removed completely and the player will still be locked out of the game. The BSG CM has a report about such wallet transactions that fail to be resolved by the Tracker. Transactions can be removed or moved to the Tracker again from that report.

The Tracker has one additional rule while tracking bets: the “Transaction not found” error code is ignored for the first 5 minutes of tracking.

Errors Within Game Clients

If errors are received in the BSG game client-server communication, the corresponding error is displayed to the player. By default, the following errors are reserved and displayed:

INTERNAL ERROR	internal error of the BSG system. The BSG development team must perform investigations.
SESSION ERROR	displayed in case of any problems reached through BSG - EC communication
Insufficient funds	if the player's balance reaches zero during the session, the player is asked to deposit more money. Corresponding dialog is displayed in the game client. If the player confirms, the EC payment page is opened (if adjusted).

By default, banks are configured to show the SESSION ERROR message within game clients for all CW errors. Also, the bank can be configured to show EC specified text within the game client for each CW error code. EC can also extend a list of error codes for their own purposes and provide BSG with the message to be shown for each new code. New error codes must be agreed upon with BSG. Messages can be localized for each supported language.

Bonuses

BSG provides two bonus functionalities that can be used by EC: cash bonus and free rounds bonuses (FRB). In case EC has their own bonuses, CW API contains workarounds to inform BSG about them.

EC System Bonus Model

The EC system can have its own bonus models. EC needs to inform BSG about bonus activities of players to allow the BSG CM to have actual reports. *Bet/Result* API response has the parameters (BONUSBET/BONUSWIN) to inform the BSG server about what part of bet/win was made with the bonus money.

BSG Cash Bonus

Cash Bonus Functionality Description

Each Cash Bonus has an initial bonus balance. The player can play BSG games allowed for the bonus (list of games can be specified) for that bonus balance. Bonus balance is persisted between game sessions (i.e. the player will continue with bonus balance. The player needs to reach the required total amount of bets (rollover amount) playing games for the bonus to release it (the rest of the bonus balance is released). It is possible to lose the Cash Bonus in case the player has played out all of the bonus balance and has not reached the rollover amount. The Cash Bonus can be configured to have an expiration period. The Cash Bonus can be configured for any BSG game.

Cash Bonuses can be awarded to players using BSG CM or using a special server-to-server award Bonus API method. BSG bonuses are available only for playing BSG games. The player can have several active bonuses simultaneously.

The following parameters need to be specified to award a bonus:	
type	used to categorize bonuses. Available types: Deposit, Slots, Loss, Prize, Promo, Special. Type does not affect functionality any way.
amount	bonus money awarded to the player. Specifies initial bonus money balance of the bonus the player will start with.
rollover multiplier	used to calculate rollover amount (sum of bets to do to release bonus to cash) = amount * rollover multiplier.
description	description for the player.
comment	comment field for support/internal use.
list of games	specifies list of games available to play for the bonus.
expiration date	bonus will expire at the beginning of the next date after the expiration date.

Additional properties of a bonus:	
time awarded	automatically filled during bonus award.
status	ACTIVE|RELEASED|LOST|CANCELLED|EXPIRED. There may also be special transitional statuses such as RELEASING|CANCELLING etc., depending on the requirements of the integration.
balance	current balance of the bonus. When a bonus is awarded - balance is set to the amount and will be used to (affected by) playing games in BONUS mode for this bonus.
bet sum	current sum of bets made playing this bonus.
comment	comment field for support/internal use.
end time	time the bonus was released/cancelled/expired/lost.
currency	Bonus has the same currency as the player (who has bonus awarded).

The player is able to play BSG games on bonus money if they have an active bonus. BSG providesa special mode for such playing: BONUS mode. To play the BONUS mode,the player needs to choose any of their active bonus and select a game to play from a list of available games for the selected bonus. Balance of the selected bonus is used to play the game. The balance of the bonus is affected by game play.

The bonus is released when the player collects the rollover amount. The current balance of the bonus is released to the player balance.

Notes about releasing:

- Bonus is released in the currency of player.
- The game will be automatically closed when the rollover amount for the current bonus is collected.
- If rollover is collected during a round at a multi-step game, the player will be able to finish their current round at the current game session. If the round was not finished and the game was closed, the player will not be able to resume the round.
- After the game session is closed, the current balance of the bonus is released to cash, all lasthands for this bonus are removed and the bonus is marked as released.

The bonus expires when the specified date is passed. If the player is playing on the bonus during expiration, the game session will be terminated.

The bonus can be canceled by an administrator from CM or by API call from ES. If the player is playing on the bonus during cancelling, the game session will be terminated.

Player can lose all money on the bonus before the rollover amount is collected. In this case, the bonus is closed with status LOST.

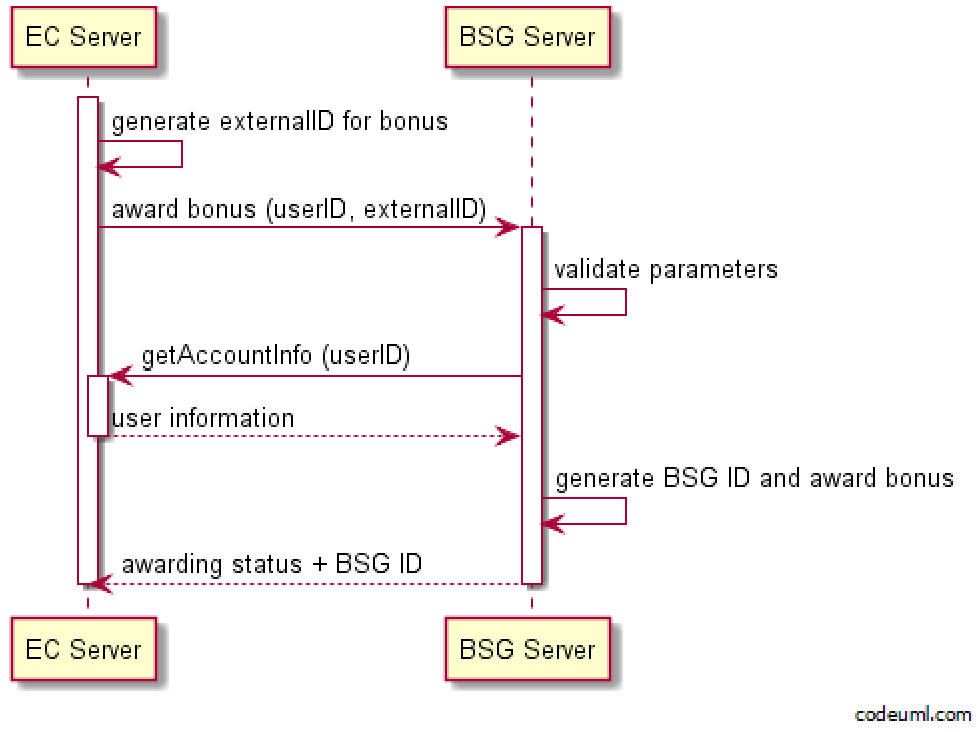
The game state during bonus games is saved separately from REAL mode games, other Cash Bonuses and FRB (see below). This means that round states are not mixed between bonuses and real mode games. The player can have unfinished rounds for the game in REAL mode and for each active bonus at the same time.

BSG provides the following API related to Cash Bonuses:	
awardBonus	allows to award a cash bonus to the player.
checkBonus	used for checking if the award Bonus was successful.
cancelBonus	allows to cancel the cash bonus.
getBonusInfo	returns information about all of the active cash bonuses of the player.
getBonusHistory	returns information about all closed cash bonuses of the player.
The EC Server must implement the following API methods to support cash bonuses:	
bonusRelease	called when a cash bonus is released, to put released money on the player's balance
getAccountInfo	called to obtain information about the player when the award Bonus API was used for a player not yet registered on the BSG server. This is very similar to Authenticate API.

Creating Bonus Using API

The EC server can create a BSG bonus for a player, using API. It allows the EC server to automatically award a bonus on an event on the EC Server (user registration, deposit, etc). Please note that it is not required to implement this. Bonuses can be also created from the BSG CM.

Awarding bonus using API diagram:



Comments:

- On the first step, the EC server should generate an external ID for a new bonus. The main purpose of this external ID is tracking the bonus. NOTE: BSG checks the external ID only among active bonuses. Closed bonuses are not checked. So this external ID does not have to be unique. It is not required to save them all. The EC server can generate an external ID, like a session for one time use for awarding.
- The EC Server calls an awardBonus API URL on the BSG server, passing user ID, external ID and all required parameters of the bonus.
- The BSG Server validates all parameters.
- The BSG Server checksplayer. If the player is not registered, BSG server makes agetAccountInfo API call to ES, receives information about the player and registers player on BSG side.
- The BSG Server creates a bonus, return status and bonus ID on the BSG side to ES.

In case ES receives error on an awardBonus API call, it is possible to check the status of an award by calling acheckBonus API, passing the externalID used during the awardBonus API call as a parameter. The BSG server responds with the bonus information if the bonus was created or responds that it does not exist.

Starting Bonus Game

Below is a common model of how a Cash Bonus should be started.

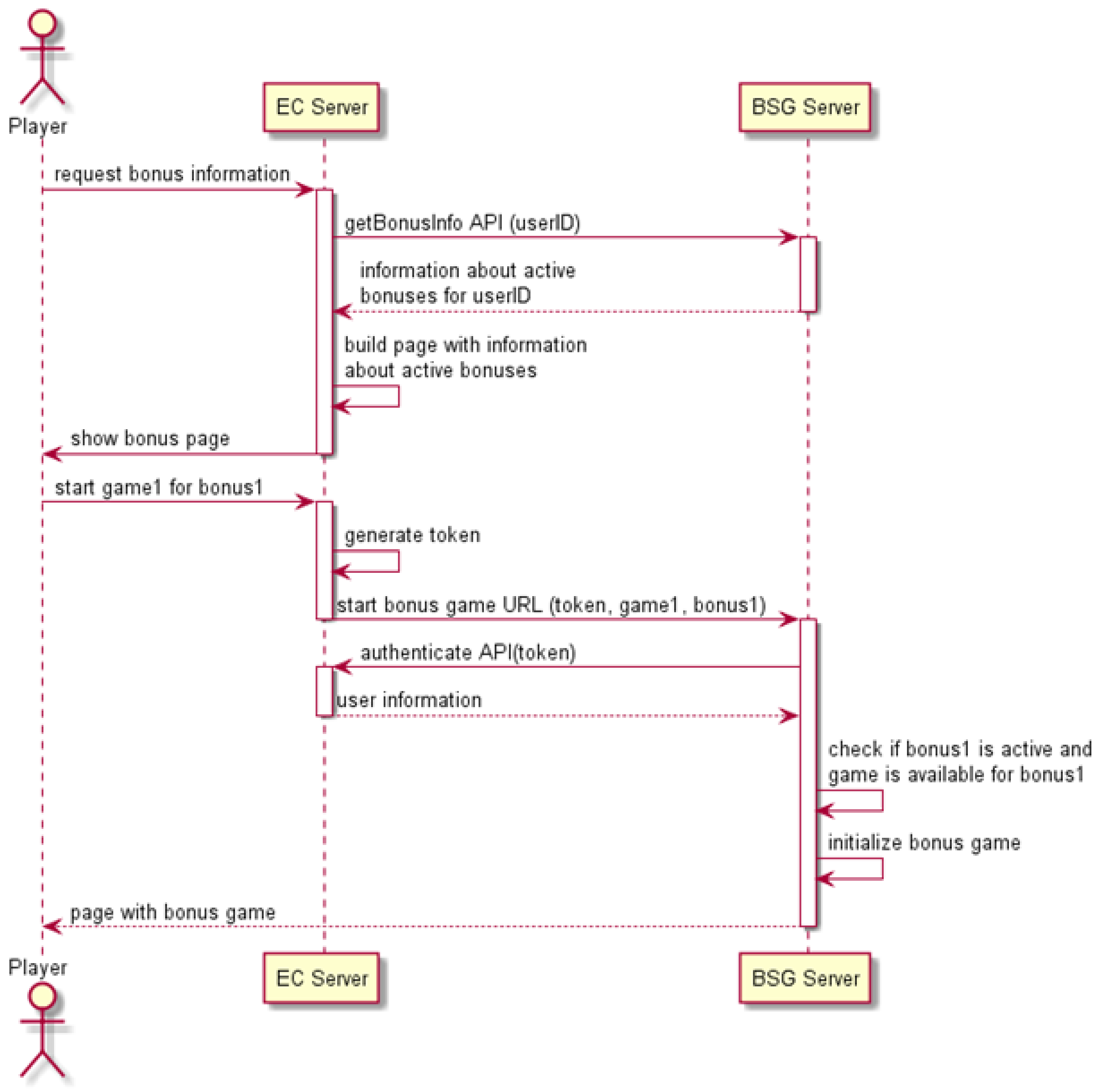
Start bonus game URL example:

/bsstartgame.do?gameId=225&bonusId=668641&token=123456&lang=en&bankId=EC1

Parameters:

token	the authentication token.
bankId	specifies from what bank the game was started. Configuration of this bank will be used to start the game.
gameId	specifies what game should be started.
bonusId	ID of the Cash Bonus the game should be started for.
lang	specifies what localization should be used.

Here is the model of how the Cash Bonus can be started by the player:



Player
EC Server
BSG Server
request bonus information
getBonusInfo API (userID)
information about active bonuses for userID
build page with information about active bonuses
show bonus page
start game1 for bonus1
generate token
start bonus game URL (token, game1, bonus1)
authenticate API(token)
user information
check if bonus1 is active and game is available for bonus1
initialize bonus game
page with bonus game
Player
EC Server
BSG Server

Description:

- EC implements a special bonus page where the player can view a list of all their active Cash Bonuses and start a bonus game. The bonus page can be built dynamically. The EC Server makes a getBonusInfo API call to the BSG Server to obtain information about all active bonuses of a player. Using the information obtained, EC renders a bonus page for the player.
- Player starts game for bonus.
- EC Server generates a token (the same way as it is described in Starting Game for Authorized Player)
- EC Server redirects the player to the BSG launch bonus game URL, providing the token as a parameter.
- The BSG server makes an authenticate API call to EC server to obtain information about the player.
- The BSG server validates parameters and starts the bonus game session for player.
- The BSG server redirects the player to the page with the game client embedded.

Free Rounds Bonus (FRB)

Description

Free Rounds Bonus (FRB) allows the player to play a defined number of free rounds in a game(s). Free rounds are always played with a defined constant bet (specified for each game per bank and currency). During FRB game play, bets/stakes are not deducted from the player's balance, but all wins received are still added to the player's balance. For each win, the BSG server sends a bonusWin API call to ES server to add money to the player's balance. Please note that not all BSG games support FRB.

Free Rounds Bonus (FRB) can be awarded to players registered within the BSG casino cluster from CM by administrators or from ES using API. The BSG FRB is available for playing BSG games only.

The following parameters need to be specified to award a FRB:	
Rounds	specifies how many free rounds the player will be able to play.
description	how many free rounds the player will be able to play.
comment	comment field for support/internal use.
list of games	specifies a list of games available to play on this bonus.
Additional properties of the bonus:	
time awarded	automatically filled during bonus award.
status	ACTIVE|CLOSED|CANCELLED.
rounds left	how many free rounds are left to be played.
end time	time the bonus was closed/cancelled.
currency	Bonus has the same currency as player (who has the bonus awarded).

The game state during FRB games is saved separately from REAL mode games, other FRB and Cash Bonuses. This means that round states are not mixed between bonuses and real mode games. The player can have unfinished rounds for the game in REAL mode and for each active bonus at the same time.

Each win received by the player during a FRB game is immediately sent to the EC server using a **bonusWin** API call. FRB is closed at the end of the game session when there are no free rounds left.

Notes about closing a FRB:

- The game will be automatically closed when all free rounds are played (if the round has several steps, the game will be automatically closed after the last step).
- In case of multistep rounds, if the last free round was interrupted (unfinished),the player will not be able to finish it. The FRB will be closed and thelasthand will be removed. This is done to not leave an active FRB with 0 free rounds left.

A FRB can be canceled by an administrator from the CM or by an API call from ES. If the player is playing on a FR bonus during cancelling, the game session will be terminated.

API Methods

BSG provides the following API related to FRB Bonuses:	
Award FRB	allows awarding FRB to player.
Check FRB	used for checking if award FRB API call was successful.
Cancel FRB	allows cancelling cash bonus.
Get FRB Info	returns information about all active cash bonuses of the player.
Get FRB History	returns information about all closed cash bonuses of the player.
The EC Server must implement the following API methods to support FRB:	
Bonus Win	called by the BSG server for each win during FRB game session.
Get Account Info	called to obtain information about the player when an award FRB API was called for a player who is not yet registered on the BSG server. Very similar to Authenticate API.

Awarding FRB Using API

The EC server can create a BSG bonus for a player using API. It allows the EC server to automatically award a FRB on some event on the EC Server (user registration, deposit, etc). Please note that it is not required to implement this. A FRB can be also created from the BSG CM.

Not all BSG games support FRB. Call the following API to obtain a list of games configured for banks and supporting FRB:

/frbgamelist.do?bankId=102

Response example:

```
<GAMESSUITES>
<SUITES>
<SUITE ID="Slots" NAME="Slots">
<GAMES>
<GAME NAME="Enchanted" ID="10207">
</GAME>
</GAMES>
</SUITE>
</SUITES>
</GAMESSUITES>
```

Awarding a FRB using API is done the same way as awarding Cash Bonuses except the API call:

- On the first step, the EC server should generate anexternalID for a new bonus. The main purpose of this external ID is to track the bonus. NOTE: BSG checks the external ID only among active bonuses. Closed bonuses are not checked, so the external ID does not have to be unique. The EC server can generate an external ID like a session for one time use for awarding.
- The EC Server calls an awardFRB API URL, passing user ID, external ID and all the required parameters of the bonus.
- The BSG Server validates all parameters.
- The BSG Server checks user. If the user is not registered, the BSG server makes a getAccountInfo API call to ES, receives information about the player and registers the player on the BSG side.
- The BSG Server creates a FRB and returns status and bonus ID to ES.

In case ES receives error on the awardFRB API call, it is possible to check the status of the award by calling acheckFRB API, passing externalID(used during the awardFRBAPI call) as a parameter. The BSG server responds with bonus information if a FRB was created or responds that it does not exist.

Starting FRB

It is possible to start a FRB using two different ways.
A FRB game is started automatically when a default REAL mode launch URL is used **/cwstartgamev2.do** and the player has an active FRB. Play mode is automatically switched to FREE ROUNDS and the game is started using the active FRB. In case the player has several active FRB, the oldest one is used first.

Another way is to start a FRB game directly. The EC can have a FRB page and start FRB games directly using special FRB launch URL:

/cwstartgameidfrb.do

Parameters:

token	
bankId	specifies from what bank game was started. Configuration of this bank will be used to start the game.
gameId	specifies which game should be started.
bonusId	ID of a Cash Bonus the game should be started for.
lang	specifies what localization should be used.

In addition, EC should use another real mode start URL to prevent a FRB from being started automatically:

/cwstartgamenotfrb.do

Parameters are the same as for ***/cwstartgamev2.do***

INTEGRATION API

Protocol Format and Description

The communication protocol is XML over HTTP/HTTPS. In order to send a request, the EC/BSG side calls specified URL with the parameters submitted using GET/POST method. The response is returned as an XML document.

The response should always be XML. All tags should be capped. The root tag name depends on the side: BSG API responds with BSGSYSTEM as the root tag, ES API responds with EXTSYSTEM as the root tag. The root tag always contains 3 tags: REQUEST, TIME and RESPONSE. The REQUEST tag contains all request parameters (as tags) which were passed in the request. TIME has value = time response was compiled. RESPONSE contains response parameters as tags. Each RESPONSE must contain a RESULT tag with possible values: OK|ERROR. OK is for a successful response, ERROR is for errors. In case of error, the RESPONSE should contain a CODE tag with code of error.

Example of a request:

http://somesite.com/awardBonus.do?userId=12312&amount=1000

Example of a response

```
< BSGSYSTEM >
<REQUEST>
<USERID>someUserID</USERID>
<AMOUNT>10000</AMOUNT>
</REQUEST>
<TIME>18 Mar 2011 12:13:44</TIME>
<RESPONSE>
<RESULT>OK</RESULT>
<BONUSID>193782</BONUSID>
</RESPONSE>
</BSGSYSTEM >
```

Security

Each API request has a "hash" parameter that contains MD5 hash (hex representation) of the string build based on request parameters and a secret PASS_KEY. The receiving side should validate the request using the hash passed (build a hash from the parameters and PASS_KEY and compare it to the passed one).

The string for hash is built by concatenating parameters in a specified order and adding PASS_KEY at the end. Values of the parameters must be exactly the same as passed in request (the values should not be reformatted). If an optional parameter has not been passed or has no value, nothing should be inserted to string at its place (do not insert "null" or "nil" or something like this).

Landing URLs on BSG Side

Start Game for Guest

Used to start the selected BSG game in free mode for an unauthorized player.

Parameters:

Name	Type	Description
gameId	Integer	ID of BSG game. List of IDs is provided by BSG.
bankId	String	ID of bank. Provided by BSG.
lang	String	Language for game client to start with. If language is not specified or not supported, the default will be used.

SUCCESS RESULT: page with game embedded ready to play in free mode.

ERROR RESULT: page with error description.

Example:
/cwguestlogin.do?bankId=128&gameId=220&lang=en

Start Game for Authorized Player

Used to start a BSG game for an authorized player.

Parameters:

Name	Type	Description
token	String	Unique token generated by ES for theplayer's current session.
gameId	Integer	ID of BSG game. List of IDs is provided by BSG.
bankId	String	ID of bank. Provided by BSG.
mode	String	Values: “real” or “free”. Specifies if the game will be played with real money or not.

lang	String	Language for the game client to start with. If language is not specified or not supported, default will be used.
homeUrl	Sting	For mobile games only: specifies URL where the player should be redirected on pressing the HOME button. In case the parameter was not passed, the default URL configured for the bank is used.
cashierUrl	String	Specifies the URL where the player should be redirected in case they have no money on their balance and the “go to cashier” button is pressed. In case the parameter was not passed, the default URL configured for the bank is used.

SUCCESS RESULT: page with the game embedded ready to play.

ERROR RESULT: page with the error description.

Example:

/cwstartgamev2.do?bankId=EC1&gameId=220&mode=real&token=123456753434&lang=en

NOTE: In case the player has an active FRB for game, the game will be started in FRB mode automatically.

Start Game for Authorized Player Without FRB Auto-start

Start BSG game for authorized player.

Parameters:

Name	Type	Description
token	String	Unique token generated by ES for the player's current session.
gameId	Integer	ID of BSG game. List of IDs is provided by BSG.
bankId	String	ID of bank. Provided by BSG.
mode	String	Values: “real” or “free”. Specifies if the game will be played with real money or not.
lang	String	Language for the game client to start with. If language is not specified or not supported, the default will be used.
homeUrl	Sting	For mobile games only: specifies URL where player should be redirected on pressing HOME button. In case parameter was not passed, the default URL configured for the bank is used.
cashierUrl	String	Specifies the URL where the player should be redirected in case they have no money on their balance and the “go to cashier” button is pressed. In case the parameter was not passed, the default URL configured for the bank is used.

SUCCESS RESULT: page with the game embedded ready to play.

ERROR RESULT: page with the error description.

Example:

/cwstartgamenotfrb.do?bankId=EC1&gameId=220&mode=real&token=123456753434&lang=en

Start Bonus Game

Used to start the selected BSG game in bonus mode for the selected bonus.

Parameters:

Name	Type	Description
token	String	Unique token generated by ES for the player's current session.
gameId	Integer	ID of BSG game. List of IDs is provided by BSG
bankId	String	ID of bank. Provided by BSG.
bonusId	Integer	ID of bonus. Provided by BSG.
lang	String	Language for the game client to start with. If language is not specified or not supported, the default will be used.

SUCCESS RESULT: page with the game embedded ready to play for the selected bonus.

ERROR RESULT: page with error description.

Example:

/bsstartgame.do?gameId=225&bonusId=668641&token=123456&lang=en&bankId=EC1

Start FRB Game

Used to start a selected BSG game in bonus mode for the selected bonus.

Parameters:

Name	Type	Description
token	String	Unique token generated by ES for player's current session.
gameId	Integer	ID of BSG game. List of IDs is provided by BSG.
bankId	String	ID of bank. Provided by BSG.
bonusId	Integer	ID of FRB.
lang	String	Language for the game client to start with. If the language is not specified or not supported, the default will be used.

SUCCESS RESULT: page with the game embedded ready to play for selected FRB.

ERROR RESULT: page with the error description.

Example:

/cwstartgameidfrb.do?gameId=225&bonusId=668641&token=123456&lang=en&bankId=EC1

Requests from BSG System to EC system

Authenticate

Authorize user by token. The method is called on stating the game/bonus for the authorized player.

Parameters:

Name	Type	Description
Token	String	Token provided by ES.
Hash	String	order:token

Response parameters:

Name	Type	Description
userId	String	Unique ID of the player within ES.
username	String	Optional
firstname	String	Optional
lastname	String	Optional
email	String	Optional
currency	String	ISO code of the player's currency. If it is not provided, the default currency configured for the bank on the BSG side will be used. (Optional)
balance	Integer	current balance of the player (in cents) on the EC system side.

Note: the firstname, lastname and email parameters are optional. The BSG system can store the information if necessary for EC convenience.

Possible error codes:

- 399   Internal Error
- 400   Invalid token
- 500   Invalid hash

Success response example:

```
< EXTSYSTEM>
        <REQUEST>
                <TOKEN>123123-12312312-12342342</TOKEN>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
                <RESULT>OK</RESULT>
                <USERID>123123</USERID>
                <USERNAME>testplayer</USERNAME>
                <FIRSTNAME>Richard</FIRSTNAME>
                <LASTNAME>Smith</LSTNAME>
                <EMAIL>rs@somemail.com</EMAIL>
                <CURRENCY>EUR</CURRENCY>
                <BALANCE>23200</BALANCE>
        </RESPONSE>
<EXTSYSTEM>
```

Error response example:

```
<EXTSYSTEM>
        <REQUEST>
<TOKEN>123123-12312312-12342342</TOKEN>
<HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
                <RESULT>ERROR</RESULT>
                <CODE>399</CODE>
        </RESPONSE>
<EXTSYSTEM>
```

Bet/Result

Request parameters:

Name	Type	Description
userId	String	Unique ID of the player within ES
bet	String	Amount of the bet (in cents) and the unique ID of the transaction on the BSG side in the format: bet_amount|transactionId
win	String	Amount of the win (in cents) and the unique ID of the transaction on the BSG side in the format: win_amount|transactionId
roundId	Integer	Unique ID of the game round onthe BSG side. The game round can have several Bet/Result requests.
gameId	Integer	Unique ID of the game.
isRoundFinished	Boolean	Optional. Informs the EC system if the current round is finished.
hash	String	Order: userId, bet, win, isRoundFinished, roundId, gameId
gameSessionId	String	Unique ID of the game session on BSG side.
negativeBet	Integer	Optional. Can be sent only with a win operation. Specifies the amount in cents to return to the player's balance as result of a partially canceled bet (can occur in *Craps* and *Ride'm Poker* games). Note: negativeBet is not part of the win amount and must be processed separately.

NOTE: Each request contains only a bet or win parameter. Bet and win cannot be sent in the same request.

The EC system must save and check the ID of BSG transactions. EC must not process the same operation (with the same ID) twice. If the transaction was already registered and successfully processed on the EC system, the EC system must return an OK response without affecting the player's balance once again.

“isRoundFinished” is an optional parameter. By default it is not passed. ES should inform BSG if it is required on the ES side to receive this flag. Please note that isRoundFinished will require sending an additional zero win operation at the end of a round, in case there was no win on the last turn (to inform ES that the round is finished).

Response parameters:

Name	Type	Description
extSystemTransactionId	String	ID of the transaction on the EC side.
balance	Integer	Balance of the player in cents on the EC system side (after transaction was applied).
bonusBet	Integer	Optional. Specifies what part of the bet was done on bonus money. If a parameter is not provided, the BSG system will account the whole bet as cash (real money).
bonusWin	Integer	Optional. Specifies what part of the win was applied to bonus money. If a parameter is not provided, the BSG system will account the whole win as cash (real money).

Possible error codes:

- `300   Insufficient funds`
- `301   Operation failed`
- `310   Unknown user id`
- `399   Internal Error`
- `500   Invalid hash`

Example of a request:

http://bsgsystem.com/betresult.do?userId=12345&bet=123407|12344546&roundId=12345&gameId=12&isRoundFinished=true&hash=8cb54b1924dbbd626a3b079a47527d17

Example of the response in case of a successful operation:

```
< EXTSYSTEM>
<REQUEST>
<USERID>12345</USERID>
<BET>123407|12344546</BET>
<ROUNDID>12345</ROUNDID>
<GAMEID>12</GAMEID>
<ISROUNDFINISHED>false</ISROUNDFINISHED>
<HASH>8cb54b1924dbbd626a3b079a47527d17</HASH>
</REQUEST>
<TIME>12 Jan 2004 15:15:15</TIME><RESPONSE>
<RESULT>OK</RESULT>
<EXTSYSTEMTRANSACTIONID>154456456</EXTSYSTEMTRANSACTIONID><B
ALANCE>235500</BALANCE>
<BONUSBET>123407</BONUSBET>
</RESPONSE>
</EXTSYSTEM>
```

Example of a response in case of a failed operation:

```
< EXTSYSTEM>
<REQUEST>
<USERID>12345</USERID>
<BET>123407|12344546</BET>
<ROUNDID>12345</ROUNDID>
<GAMEID>12</GAMEID>
<ISROUNDFINISHED>false</ISROUNDFINISHED>
<HASH>8cb54b1924dbbd626a3b079a47527d17</HASH>
</REQUEST>
<TIME>12 Jan 2004 15:15:15</TIME><RESPONSE>
<RESULT>FAILED</RESULT>
<CODE>300</CODE>
</RESPONSE>
</EXTSYSTEM>
```

Refund Bet

Request parameters:

Name	Type	Description
userId	String	Unique ID of the player within ES
casinoTransactionId	String	ID of the transaction on the BSG side. ID of the transaction is sent to the EC system with BetResult call.
hash	String	order: userId, casinoTransactionId

Response parameters:

Name	Type	Description
extSystemTransactionId	String	ID of the transaction on the EC side.

Possible error codes:

- `302    Unknown transaction id`
- `310    Unknown user id`
- `399    Internal Error`
- `500    Invalid hash`

Example of a request:

http://bsgsystem.com/refundBet.do?userId=12345&casinoTransactionId=12344546&hash=8cb54b1924dbbd626a3b079a47527d17

Example of a response in case of successful operation:

```
< EXTSYSTEM>
<REQUEST>
<USERID>12345</USERID>
<CASINOTRANSACTIONID>123787845</CASINOTRANSACTIONID>
<HASH>8cb54b1924dbbd626a3b079a47527d17</HASH>
</REQUEST>
<TIME>12 Jan 2004 15:15:15</TIME><RESPONSE>
<RESULT>OK</RESULT>
<EXTSYSTEMTRANSACTIONID>154456456</EXTSYSTEMTRANSACTIONID>
</RESPONSE>
</EXTSYSTEM>
```

Example of a response in case of a failed operation:

```
< EXTSYSTEM>
<REQUEST>
<USERID>12345</USERID>
<CASINOTRANSACTIONID>123787845</CASINOTRANSACTIONID>
<HASH>8cb54b1924dbbd626a3b079a47527d17</HASH>
</REQUEST>
<TIME>12 Jan 2004 15:15:15</TIME>
<RESPONSE>
<RESULT>FAILED</RESULT>
<CODE>301</CODE>
</RESPONSE>
</EXTSYSTEM>
```

Get Balance

Returns the actual balance of the player on the EC system.This can be used in case the game lobby is implemented on the BSG side.

Request parameters:

Name	Type	Description
userId	String	Unique ID of the user within ES

Response parameters:

Name	Type	Description
balance	Integer	Balance of the player in cents on EC system side.

Possible error codes:

- `310    Unknown user id`
- `399    Internal Error`

Example of a request:

http://bsgsystem.com/getbalance.do?userId=12345

Example of a response in case of successful operation:

```
< EXTSYSTEM>
<REQUEST>
<USERID>12345</USERID>
</REQUEST>
<TIME>12 Jan 2004 15:15:15</TIME><RESPONSE>
<RESULT>OK</RESULT>
<BALANCE>12300</BALANCE>
</RESPONSE>
</EXTSYSTEM>
```

Example of a response in case of a failed operation:

```
< EXTSYSTEM>
<REQUEST>
<USERID>12345</USERID>
</REQUEST>
<TIME>12 Jan 2004 15:15:15</TIME>
<RESPONSE>
<RESULT>FAILED</RESULT>
<CODE>310</CODE>
</RESPONSE>
</EXTSYSTEM>
```

Get Account Info

Gets account info by the ES ID of the player. This is called in case BSG got an API call for an unregistered user.

Parameters:

Name	Type	Description
userId	String	Unique ID of the user within ES
hash	String	order: userId

Response parameters:

Name	Type	Description
Username	String	Optional
Firstname	String	Optional
Lastname	String	Optional
Email	String	Optional
Currency	String	ISO code of the player's currency. If it is not provided, the default currency configured for the bank on the BSG side will be used. Optional.

Possible error codes:

- `310    Unknown user id`
- `399    Internal Error`
- `500    Invalid hash`

Example of a request:

http://bsgsystem.com/accountinfo.do?userId=432&hash=1e3b0ae551b1dfdc48137bc50ad26d1c

Success response example:

```
< EXTSYSTEM>
        <REQUEST>
                <USERID>432</USERID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
                <RESULT>OK</RESULT>
                <USERNAME>testplayer</USERNAME>
                <FIRSTNAME>Richard</FIRSTNAME>
                <LASTNAME>Smith</LSTNAME>
                <EMAIL>rs@somemail.com</EMAIL>
                <CURRENCY>EUR</CURRENCY>
        </RESPONSE>
<EXTSYSTEM>
```

Error response example:

```
< EXTSYSTEM>
        <REQUEST>
                <USERID>432</USERID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
                <RESULT>ERROR</RESULT>
                <CODE>500</CODE>
        </RESPONSE>
<EXTSYSTEM>
```

Bonus Release

BSG sends this request to ES to release a bonus to the player's cash balance. It is assumed that this operation cannot fail. In case of error code 699 or a network error, the bonusRelease call will be repeated with the same parameters. If ES has the bonus already released (check by bonusId), then ES should just reply with an OK response. Other errors will require manual resolving.

Parameters:

Name	Type	Description
userId	String	Unique ID of the player within ES.
bonusId	Integer	ID of the bonus on BSG side.
amount	Integer	Amount in cents to release to cash.
hash	String	Order: userId, bonusId, amount

Possible error codes:

- `310    Unknown user id`
- `399    Internal Error`
- `500    Invalid hash`

Example of a request:

http://bsgsystem.com/bonusrelease.do?userId=123423&bonusId=32132&amount=7820&hash=1e3b0ae551b1dfdc48137bc50ad26d1c

Success response example:

```
<EXTSYSTEM>
        <REQUEST>
                <USERID>123423</USERID>
                <BONUSID>32132</BONUSID>
                <AMOUNT>7820</AMOUNT>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
                <RESULT>OK</RESULT>
        </RESPONSE>
<EXTSYSTEM>
```

Error response example:

```
EXTSYSTEM>
	<REQUEST>
		<USERID>123423</USERID>
		<BONUSID>32132</BONUSID>
		<AMOUNT>7820</AMOUNT>
		<HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
	</REQUEST>
	<TIME>18 Mar 2011 12:13:44</TIME>
	<RESPONSE>
		<RESULT>ERROR</RESULT>
		<CODE>699</CODE>
	</RESPONSE>
<EXTSYSTEM>
```

Bonus Win

BSG sends this request for each win obtained during a FRB game session. All wins should be added to the player's cash balance. It isassumed that this operation cannot fail. In case of error code 699 or a network error, the bonusWin call will be repeated with the same parameters. If ES has the bonusWin already received and successfully processed (check by transactionId), then ES should reply with OK response. Other errors will require manual resolving.

Parameters:

Name	Type	Description
userId	String	ID of used on ES side.
bonusId	Integer	ID of the FR bonus on the BSG side.
amount	Integer	Amount of the win in cents.
transactionId	String	ID of the transaction on the BSG side
hash	String	Order: userId, bonusId, amount

Response parameters:

Name	Type	Description
balance	Integer	Balance of the player (in cents) on ES side after applying a win.

Possible error codes:

```
602   Invalid or empty amount
610   Invalid parameters
620   Invalid hash
631   User not found
699   Internal error
```

Example of a request:

http://bsgsystem.com/bonuswin.do?userId=123423&bonusId=32132&amount=7820&transactionId=7820&hash=1e3b0ae551b1dfdc48137bc50ad26d1c

Success response example:

```
<EXTSYSTEM>
        <REQUEST>
                <USEDID>123423</USEDID>
                <BONUSID>32132</BONUSID>
                <AMOUNT>7820</AMOUNT>
                <TRANSACTIONID>123456789</TRANSACTIONID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
        <RESULT>OK</RESULT>
        <BALANCE>12345</BALANCE>
        </RESPONSE>
</EXTSYSTEM>
```

Error response example:

```
<EXTSYSTEM>
        <REQUEST>
                <USEDID>123423</USEDID>
                <BONUSID>32132</BONUSID>
                <AMOUNT>7820</AMOUNT>
                <TRANSACTIONID>123456789</TRANSACTIONID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
        <RESULT>ERROR</RESULT>
        <CODE>699</CODE>
        </RESPONSE>
</EXTSYSTEM>
```

Bonus API on BSG side

Award Bonus

Used to award a bonus. In case of a network error, the status of the award can be checked usinga checkBonus API call.

Parameters:

Name	Type	Description
userId	String	Unique ID of the player within ES.
bankId	String	ID of bank. Provided by BSG.
type	String	Type of bonus. Values: Deposit, Slots, Loss, Prize, Promo, Special
amount	integer	Amount of bonus in cents.
multiplier	float	Multiply to calculate the rollover amount.
games	String	Values: all|except|only
gameIds	String	List of BSG game IDs separated with "|"
expDate	String	Date in format DD.MM.YYYY
comment	String	Optional comment field for internal use.
description	String	Optional description for the bonus to show to the player.
extBonusId	String	Bonus ID from ES.
hash	String	Order: userId, bankId, type, amount, multiplier, games, gameIds, extDate, comment, description, extBonusId

Response parameters:

Name	Type	Description
bonusId	integer	ID of the bonus on the BSG side.

Possible error codes:

- 601 Invalid or empty bonus type
- 602 Invalid or empty amount
- 603 Invalid or empty multiplier
- 604 Invalid or empty game modes
- 605 Invalid or empty game ID
- 606 Invalid or empty exp Date
- 610 Invalid parameters
- 620 Invalid hash
- 699 Internal error
- 641 Bonus with such extBonusId already exists

Success response example:

```
<BSGSYSTEM>
        <REQUEST>
                <USERID>someUserID</USERID>
                <TYPE>Slot</TYPE>
                <AMOUNT>10000</AMOUNT>
                <MULTIPLIER>10.0</MULTIPLIER>
                <GAMES>all</GAMES>
                <EXPDATE>01.05.2011</EXPDATE>
        <EXTBONUSID>123423</EXTBONUSID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
                <RESULT>OK</RESULT>
                <BONUSID>193782</BONUSID>
        </RESPONSE>
</BSGSYSTEM>
```

Error response example:

```
<BSGSYSTEM>
        <REQUEST>
                <USERID>someUserID</USERID>
                <TYPE>Slot</TYPE>
                <AMOUNT>10000</AMOUNT>
                <MULTIPLIER>10.0</MULTIPLIER>
                <GAMES>all</GAMES>
                <EXPDATE>01.05.2011</EXPDATE>
        <EXTBONUSID>123423</EXTBONUSID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
                <RESULT>ERROR</RESULT>
                <CODE>611</CODE>
        </RESPONSE>
</BSGSYSTEM>
```

Check Bonus

Used to check the status of a bonus award in case of a network error during the award Bonus call.

Parameters:

Name	Type	Description
extBonusId	String	Bonus ID from ES.
bankId	String	ID of bank. Provided by BSG.
hash	String	Order: extBonusId, bankId

Response parameters:

Name	Type	Description
bonusId	integer	ID of the bonus on the BSG side.

Possible error codes:

- `610    Invalid parameters`
- `620    Invalid hash`
- `630    Operation not found`
- `699    Internal error`

Success response example:

```
<BSGSYSTEM>
        <REQUEST>
<EXTBONUSID>123423</EXTBONUSID>
<HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
                <RESULT>OK</RESULT>
                <BONUSID>193782</BONUSID>
        </RESPONSE>
</BSGSYSTEM>
```

Error response example:

```
<BSGSYSTEM>
        <REQUEST>
<EXTBONUSID>123423</EXTBONUSID>
<HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
                <RESULT>ERROR</RESULT>
                <CODE>630</CODE>
        </RESPONSE>
</BSGSYSTEM>
```

Cancel Bonus

Parameters:

Name	Type	Description
bonusId	Integer	ID of the bonus on the BSG side
hash	String	Order: bonusId

Possible error codes:

- `610    Invalid parameters`
- `620    Invalid hash`
- `630    Operation not found`
- `699    Internal error`

Success response example:

```
<BSGSYSTEM>
        <REQUEST>
                <BONUSID>123423</BONUSID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
                <RESULT>OK</RESULT>
        </RESPONSE>
</BSGSYSTEM>
```

Error response example:

```
<BSGSYSTEM>
        <REQUEST>
                <BONUSID>123423</BONUSID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
                <RESULT>ERROR</RESULT>
                <CODE>630</CODE>
        </RESPONSE>
```

Get Bonus Info

Returns active bonuses for the player.

Parameters:

Name	Type	Description
userId	String	Unique ID of the player within ES.
bankId	String	ID of bank. Provided by BSG.
hash	String	Order: userId, bankId

Response parameters:

Name	Type	Description
bonus		Contains info about the bonus.

The following tags are specified for each bonus:

- **BONUS ID**
  ID of the bonus on the BSG side
- **TYPE**
  Type of bonus. Possible values: Deposit, Slots, Loss, Prize, Promo, Special
- **AWARDED DATE**
  Date the bonus was awarded. Format: DD.MM.YYYY
- **AMOUNT**
  Initial amount of the bonus in cents.
- **BALANCE**
  Current balance of the bonus in cents.
- **ROLLOVER**
  Rollover amount in cents.
- **COLLECTED**
  Currently collected rollover amount in cents.
- **DESCRIPTION**
  Description of the bonus, entered on awarding.

- **GAMEIDS**
  Comma-delimited list of BSG IDs of games available to play the bonus.
- **EXPDATE**
  Date the bonus will expire. Format: DD.MM.YYYY

Possible error codes:

- `610   Invalid parameters`
- `620   Invalid hash`
- `699   Internal error`

Success response example:

```
< BSGSYSTEM >
        <REQUEST>
<USERID>111</USERID>
<HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
                <RESULT>OK</RESULT>
                <BONUS>
                        <BONUSID>123</BONUSID>
                        <TYPE>Slot</TYPE>
                        <AWARDEDDATE>01.01.2011</ AWARDEDDATE>
                        <AMOUNT>10000</AMOUNT>
                        <BALANCE>9723</BALANCE>
                        <ROLLOVER>100000</ROLLOVER>
                        <COLLECTED>15000</COLLECTED>
                        <DESCRIPTION>Mega Slot Bonus</DESCRIPTION>
                        <GAMEIDS>5,10,222,211</GAMEIDS>
                        <EXPDATE>01.04.2011</EXPDATE>
                </BONUS>
                <BONUS>
                        <BONUSID>124</BONUSID>
                        <TYPE>Deposit</TYPE>
                        <AWARDEDDATE>01.01.2011</ AWARDEDDATE>
                        <AMOUNT>5000</AMOUNT>
                        <BALANCE>6395</BALANCE>
                        <ROLLOVER>50000</ROLLOVER>
                        <COLLECTED>43405</COLLECTED>
                        <DESCRIPTION>First Deposit Bonus</DESCRIPTION>
                        <GAMEIDS>5,10,222,211</GAMEIDS>
                        <EXPDATE>25.06.2011</EXPDATE>
                </BONUS>
        </RESPONSE>
</BSGSYSTEM >
```

Error response example:

```
<BSGSYSTEM >
        <REQUEST>
                <USERID>123423</USERID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
                <RESULT>ERROR</RESULT>
                <CODE>620</CODE>
        </RESPONSE>
</BSGSYSTEM >
```

Get Bonus History

Returns a list of finished bonuses.

Parameters:

Name	Type	Description
userId	String	Unique ID of the player within ES.
bankId	String	ID of bank. Provided by BSG.
hash	String	Order: userId, bankId

Response parameters:

Name	Type	Description
bonus		Contains info about the bonus.

The following tags are specified for each bonus:

- **BONUS ID**
  ID of the bonus on the BSG side.
- **STATUS**
  RELEASED|EXPIRED|CANCELLED|LOST
- **TYPE**
  Type of bonus. Possible values: Deposit, Slots, Loss, Prize, Promo, Special
- **AWARDED DATE**
  Date the bonus was awarded. Format: DD.MM.YYYY
- **AMOUNT**
  Initial amount of the bonus in cents.
- **BALANCE**
  Balance of the bonus in cents at the moment it was closed.
- **ROLLOVER**
  Rollover amount in cents.
- **COLLECTED**
  Collected amount in cents at the moment it was closed.
- **DESCRIPTION**
  Description of thebonus, entered on awarding.
- **GAME IDS**
  Comma-delimited list of BSG IDs of games available to play the bonus.
- **END DATE**
  Date the bonus was closed. Format: DD.MM.YYYY

Possible error codes:

- `610   Invalid parameters`
- `620   Invalid hash`
- `699   Internal error`

Success response example:

```
<BSGSYSTEM>
	<REQUEST>
<USERID>111</USERID>
<HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
	</REQUEST>
	<TIME>18 Mar 2011 12:13:44</TIME>
	<RESPONSE>
		<RESULT>OK</RESULT>
		<BONUS>
			<BONUSID>123</BONUSID>
			<STATUS>RELEASED</STATUS>
			<TYPE>Slot</TYPE>
			<AWARDEDDATE>01.01.2011</AWARDEDDATE>
			<AMOUNT>10000</AMOUNT>
			<BALANCE>9723</BALANCE>
			<ROLLOVER>100000</ROLLOVER>
			<COLLECTED>100025</COLLECTED>
			<DESCRIPTION>Mega Slot Bonus</DESCRIPTION>
			<GAMEIDS>5,10,222,211</GAMEIDS>
			<ENDDATE>01.04.2011</ENDDATE>
		</BONUS>
	</RESPONSE>
</BSGSYSTEM>
```

Error response example:

```
<BSGSYSTEM>
	<REQUEST>
		<USERID>123423</USERID>
		<HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
	</REQUEST>
	<TIME>18 Mar 2011 12:13:44</TIME>
	<RESPONSE>
		<RESULT>ERROR</RESULT>
		<CODE>620</CODE>
	</RESPONSE>
</BSGSYSTEM>
```

FRB API on BSG side

Award FR Bonus

Used to award FRB. In case of a network error, the status of an award can be checked using a checkFRBonus API call.

Parameters:

Name	Type	Description
userId	String	Unique ID of user within ES.
bankId	String	ID of the bank. Provided by BSG. Optional.
rounds	integer	Available free rounds count.
games	String	List of BSG game IDs separated with "|"
comment	String	Optional comment for internal use.
description	String	Optional description for the bonus to show to the player.
extBonusId	String	Bonus ID from ES.
startTime	String	Optional. The time the bonus will begin to be available to the player. BSG server time. Format: dd.MM.yyyyHH:mm:ss or dd.MM.yyyy If not specified, it will be available immediately after awarded.
expirationTime	String	Optional. The time bonus will expire. BSG server time. Format: dd.MM.yyyyHH:mm:ss or dd.MM.yyyy If not specified, will not expire ever.
duration	integer	Optional. Duration in days that the bonus will be available to player from the moment of first time bonus was used by player. If not specified, there is no duration limit.

expirationHours	integer	Optional. If used with expirationTime, expirationTime will be ignored. If used with startTime will be calculated expirationTime as startTime+expirationHours. If expirationHours used without startTime and expirationTime will calculated expirationTime as current system time + expirationHours.
frbTableRoundChips	integer	Optional. Used when FRB is awarded for table games. Specifies the balance, in cents, available for the player for each FRB round. In case a value will not be provided, the default (configured for bank) will be used.
hash	String	Order: usedId, bankId, rounds, games, comment, description, extBonusId

Response parameters:

Name	Type	Description
bonusId	integer	ID of the bonus on the BSG side.

Possible error codes:

```
605   Invalid or empty games
607   Invalid rounds
610   Invalid parameters
620   Invalid hash
631   Invalid user
641   Bonus with such ExtBonusId specified already exists
642   Invalid Duration
643   Invalid startTime
644   Invalid expirationTime
645   Expiration Time less or equals Start Time
646   Duration is greater than selected time period
647   Invalid expirationHours
648   Invalid table chips value
699   Internal error
```

Success response example:

```
<BSGSYSTEM >
            <REQUEST>
            <USERID>someUserID</USERID>
            <ROUNDS>20</ROUNDS>
                        <GAMES>224|225|226</GAMES>
                        <EXPDATE>01.05.2011</EXPDATE>
            <EXTBONUSID>123423</EXTBONUSID>
                        <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
            </REQUEST>
            <TIME>18 Mar 2011 12:13:44</TIME>
            <RESPONSE>
            <RESULT>OK</RESULT>
            <BONUSID>193782</BONUSID>
            </RESPONSE>
</BSGSYSTEM >
```

Error response example:

```
<BSGSYSTEM >
            <REQUEST>
            <USERID>someUserID</USERID>
            <ROUNDS>20</ROUNDS>
                        <GAMES>224|225|226</GAMES>
                        <EXTBONUSID>123423</EXTBONUSID>
                        <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>

            </REQUEST>
            <TIME>18 Mar 2011 12:13:44</TIME>
            <RESPONSE>
            <RESULT>ERROR</RESULT>
            <CODE>611</CODE>
            </RESPONSE>

</BSGSYSTEM >
```

Check Bonus

Used to check the status of a bonus award in case of a network error during an awardFRBonus call.

Parameters:

Name	Type	Description
extBonusId	String	Bonus ID from ES.
bankId	String	ID of bank. Provided by BSG. Optional.
hash	String	Order: extBonusId, bankId

Response parameters:

Name	Type	Description
bonusId	integer	ID of the bonus on the BSG side.

Possible error codes:

610	Invalid parameters
620	Invalid hash
630	Bonus not found
699	Internal error

Success response example:

```
<BSGSYSTEM >
        <REQUEST>
                <EXTBONUSID>123423</EXTBONUSID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
        <RESULT>OK</RESULT>
        <BONUSID>193782</BONUSID>
        </RESPONSE>
</BSGSYSTEM >
```

Error response example:

```
<BSGSYSTEM >
        <REQUEST>
                <EXTBONUSID>123423</EXTBONUSID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>

        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
        <RESULT>ERROR</RESULT>
        <CODE>630</CODE>
        </RESPONSE>

</BSGSYSTEM >
```

Cancel Bonus

Parameters:

Name	Type	Description
bonusId	Integer	ID of the bonus on the BSG side.
hash	String	Order: bonusId

Possible error codes:

```
610   Invalid parameters
620   Invalid hash
630   Bonus not found
699   Internal error
```

Success response example:

```
<BSGSYSTEM >
        <REQUEST>
                <EXTBONUSID>123423</EXTBONUSID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
        <RESULT>OK</RESULT>
        </RESPONSE>
</BSGSYSTEM >
```

Error response example:

```
<BSGSYSTEM >
        <REQUEST>
                <EXTBONUSID>123423</EXTBONUSID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
        <RESULT>ERROR</RESULT>
         <CODE>630</CODE>
        </RESPONSE>
</BSGSYSTEM >
```

Get FR Bonus Info

Returns active FR bonuses for the player.

Parameters:

Name	Type	Description
userId	String	Unique ID of the player within ES.
bankId	String	ID of bank. Provided by BSG. Optional.
hash	String	Order: userId, bankId

Response parameters:

Name	Type	Description
bonus		Contains info about bonus

The following tags are specified for each bonus:

- **BONUSID**
  ID of the bonus on the BSG side.
- **AWARDEDDATE**
  Date the bonus was awarded. Format: DD.MM.YYYY
- **ROUNDS**
  Initial FR count.
- **ROUNDSLEFT**
  Number of free rounds left.
- **GAMEIDS**
  "|"-delimited list of BSG IDs of games available to play the bonus.
- **DESCRIPTION, STARTTIME, EXPIRATIONTIME, DURATION** – shown in case these properties were specified on award.

Possible error codes:

```
610   Invalid parameters
620   Invalid hash
631   Invalid user
699   Internal error
```

Success response example:

```
< BSGSYSTEM >
        <REQUEST>
                <USERID>111</USERID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
        <RESULT>OK</RESULT>
        <BONUS>
                <BONUSID>123</BONUSID>
                <AWARDEDDATE>01.01.2011</AWARDEDDATE>
                <ROUNDS>20</ROUDNS>
                <ROUNDSLEFT>12</ROUNDSLEFT>
                <GAMEIDS>225|226</GAMEIDS>
        </BONUS>
        <BONUS>
                <BONUSID>124</BONUSID>
                <AWARDEDDATE>01.01.2011</ AWARDEDDATE >
                <ROUNDS>15</ROUDNS>
                <ROUNDSLEFT>5</ROUNDSLEFT>
                <GAMEIDS>225</GAMEIDS>
                <DESCRIPTION>PPP SPECIAL</DESCRIPTION>
                <STARTTIME>01.01.2011</STARTIME>
                        <EXPIRATIONTIME>30.05.2012
                        00:00:00</EXPIRATIONTIME>
                        <DURATION>1</DURATION>
        </BONUS>
        </RESPONSE>
</BSGSYSTEM >
```

Success response example:

```
< BSGSYSTEM >
        <REQUEST>
                <USEDID>123423</USERID>
                <HASH>1e3b0ae551b1dfdc48137bc50ad26d1c</HASH>
        </REQUEST>
        <TIME>18 Mar 2011 12:13:44</TIME>
        <RESPONSE>
        <RESULT>ERROR</RESULT>
        <CODE>620</CODE>
        </RESPONSE>
</BSGSYSTEM >
```

FAQ

Game is not started.

“This game does not exist“ error message on game launch

The game was not configured for the bank. Please check that the bank and gameId parameters contain correct values. A list of games configured for the bank can be obtained using the Game List API URL.

“Bank is incorrect” error message on game launch

Please ensure that the value of the bankId parameter is correct. Refer to integration information provided by BSG during system configuration.

“Game disabled” error message on game launch

Specified game is disabled. Please check that the bank and gameId parameters contain correct values. If all is correct, please contact BSG support to receive comments about the reason and duration that the game was disabled.

“Internal error” error message on game launch

Possible reasons:

- The BSG server receives an error from the authenticate API for the provided token. Please check the logs for Authenticate API calls from the BSG server for the provided token.
- The BSG server has pending wallet operation for the player and game that can’t be resolved. Please check the logs for Bet/Result or Refund Bet API calls for the user and the game specified in launch URL.
- Internal error on BSG server. Please contact BSG support with the issue.

Error in game after first spin

Most likely the BSG server has received an error from the EC server on the Bet/Result API call. Please check the logs for error. If no errors are found, contact BSG support with the issue.

Wrong balance in game

BSG does not manage balances. BSG have screenshots of the balance received from the EC server on API calls. Please check the BALANCE tag in responses to BSG.

Session expired

By default, the session is expired after 10 minutes of inactivity (this can be configured to another value at the EC’s request). Another reason is that the session was closed from the CM (killed) or another session for the same player was started.

Two or more games simultaneously

BSG does not support playing 2 or more games simultaneously. The player can have only one session and game session at the same time.

Currency of player

The player can have only one single currency on the BSG side. Currency is set for the player at the moment the game is first started (from Authenticate API call) or on a bonus award using API (GetAccountInfo API call) or on a mass bonus award (specified in csv). The BSG server does not support the changing of currency of the player. Currency can be changed at the EC's request to BSG support. Please also note that changing currency of an active player (who has played several games) is time consuming because it requires time to rebuild data in CM.

Default currency of bank

Default currency can be changed at the EC's request. Please also see note about changing the currency of players above.

The same user is registered twice in BSG CM.

UserID is case sensitive for the BSG server. In case the EC server sends it differently, the BSG server creates duplicate players.

Game ID of mobile games.

By default, the BSG server separates games by platforms. I.e. Mr. Vegas, Mr. Vegas Android, Mr. Vegas Mobile, Mr. Vegas Windows Phone – all have their own game ID and are denoted as separate games (with their own PJ banks, own pending rounds etc). At the EC's request, the BSG server can be configured to use one single gameID for the game for all platforms.

Several Refund Bet API calls

The BSG server ignores error code = 302 from a Refund Bet API for 5 minutes, in case the original Bet/Result API resulted in timeout or another non-protocol error.

Changing game configurations

Game configuration (such as limit, coins, default bet etc.) can be changed by contacting BSG support.

Changing FRB configuration

The FRB configuration for a game (coin, bet per line, number of lines used during FRB) can be changed at the EC's request.

APPENDIX A - RESERVED ERROR CODES

BSG system error codes:	
Code	**Description**
601	Invalid or empty bonus type
602	Invalid or empty amount
603	Invalid or empty multiplier
604	Invalid or empty game modes
605	Invalid or empty game ID
606	Invalid or empty exp Date
607	Invalid rounds parameter
610	Invalid parameters
620	Invalid hash
630	Bonus not found
631	Invalid user
641	Bonus with such extBonusId already exists
642	Invalid duration
643	Invalid start time
644	Invalid expiration time
645	Expiration time less or equals to start time
646	Duration is greater than selected period
647	Invalid or empty expirationHours
699	Internal error

ES error codes:	
Code	**Description**
300	Insufficient funds
301	Operation failed
302	Unknown transaction id
310	Unknown user id
399	Internal Error
400	Invalid token
500	Invalid hash