

**Standing Stone Gaming**  
**Open Gaming Protocol (OGP)**  
**Specification**

**Version 2.3**



**Revised: October 1, 2003**

**Prepared for:**  
**Standing Stone Gaming**  
**Verona, NY**

**Prepared by:**  
**Rite-Solutions, Inc.**  
**Middletown, RI**

## Table of Contents

<b>1</b>	<b>SCOPE.....</b>	<b>4</b>
<b>1.1</b>	<b>Identification .....</b>	<b>4</b>
<b>1.2</b>	<b>Overview .....</b>	<b>4</b>
<b>1.3</b>	<b>Compliance .....</b>	<b>4</b>
<b>1.4</b>	<b>Document Organization .....</b>	<b>6</b>
<b>2</b>	<b>OGP BASICS AND DOCUMENT CONVENTIONS .....</b>	<b>7</b>
<b>2.1</b>	<b>Data Types and Sizes .....</b>	<b>7</b>
2.1.1	Notation.....	7
2.1.2	Integer Values .....	7
2.1.3	Character Sequences (Strings) .....	7
<b>2.2</b>	<b>Font Conventions .....</b>	<b>8</b>
<b>2.3</b>	<b>Message Definitions .....</b>	<b>8</b>
<b>3</b>	<b>OGP MESSAGE FORMATS.....</b>	<b>10</b>
<b>3.1</b>	<b>General Message Format.....</b>	<b>10</b>
3.1.1	Start of Message Word.....	10
3.1.2	Message Size Word.....	10
3.1.3	Message Contents .....	11
3.1.4	Message Checksum Word.....	11
3.1.5	End of Message Word.....	11
<b>3.2</b>	<b>Game Station Originated Messages.....</b>	<b>11</b>
3.2.1	Game Station Boot Message (OGP_GSTP_GSBOOT) .....	12
3.2.2	Game Station Status Message (OGP_GSTP_STATUS) .....	12
3.2.3	Player Logon Request Message (OGP_GSTP_LOGON) .....	13
3.2.4	Player Logoff Request Message (OGP_GSTP_LOGOFF) .....	14
3.2.5	Game Purchase Request Message (OGP_GSTP_PURCHREQ) .....	15
3.2.6	Game Play Message (OGP_GSTP_GAME_PLAY) .....	15
3.2.7	User Preferences Message (OGP_GSTP_USERPREF) .....	17
3.2.8	Game Station Statistics Message (OGP_GSTP_GS_STATS).....	17
3.2.9	Acknowledge Message (OGP_GSTP_ACK) .....	18
3.2.10	ATM Request Message (OGP_GSTP_ATM_REQUEST) .....	19
3.2.11	Player Rating Message (OGP_GSTP_RATING) .....	20
<b>3.3</b>	<b>Transaction Processing System Originated Messages.....</b>	<b>20</b>
3.3.1	Status Request Message (OGP_TPGS_STATUSREQ) .....	21
3.3.2	Player Logoff Message (OGP_TPGS_LOGOFF).....	21
3.3.3	Purchase Message (OGP_TPGS_PURCH).....	22
3.3.4	Player Logon Response Message (OGP_TPGS_LOGON).....	23

---

3.3.5	Game Result Message (OGP_TPGS_GAME_RESULT) .....	24
3.3.6	Game Station Statistics Request Message (OGP_TPGS_GS_STATSREQ).....	24
3.3.7	Acknowledge Message (OGP_TPGS_ACK).....	25
3.3.8	Game Station Enable Message (OGP_TPGS_ENABLE).....	26
3.3.9	Game Station Disable Message (OGP_TPGS_DISABLE) .....	27
3.3.10	Game Station Configuration Message (OGP_TPGS_GSCONFIG).....	27
3.3.11	Clock Synchronization Message (OGP_TPGS_TIME).....	28
3.3.12	Progressive Payout Message (OGP_TPGS_PROGRESSIVE) .....	29
3.3.13	Terminal Notify Message (OGP_TPGS_TERMNOTIFY) .....	30
3.3.14	Account Balance Update Message (OGP_TPGS_ACCT_UPDATE) .....	31
3.3.15	Draw Service Message (OGP_TPGS_DRAWSERVICE) .....	31
<b>4</b>	<b>OGP BYTE STREAMS .....</b>	<b>33</b>
<b>4.1</b>	<b>Byte Stream Definition Format .....</b>	<b>33</b>
<b>4.2</b>	<b>OGP Byte Stream Definitions .....</b>	<b>34</b>
4.2.1	Start of Message Byte Stream ( <i>START_MSG_BYTE_STREAM</i> ) .....	34
4.2.2	Message Header Byte Stream ( <i>MSG_HEADER_BYTE_STREAM</i> ).....	34
4.2.3	End of Message Byte Stream ( <i>END_MSG_BYTE_STREAM</i> ).....	36
4.2.4	Account Number Byte Stream ( <i>ACCTNO_BYTE_STREAM</i> ) .....	36
4.2.5	Account Information Byte Stream ( <i>ACCT_INFO_BYTE_STREAM</i> ) .....	37
4.2.6	Account Update Byte Stream ( <i>ACCT_UPDT_BYTE_STREAM</i> ).....	38
4.2.7	User Preferences Byte Stream ( <i>USER_PREF_BYTE_STREAM</i> ).....	39
4.2.8	Purchase Item Request Byte Stream ( <i>PURCH_ITEM_REQ_BYTE_STREAM</i> ) .....	40
4.2.9	Purchase Request Byte Stream ( <i>PURCH_REQ_BYTE_STREAM</i> ) .....	41
4.2.10	Purchase Item Complete Byte Stream ( <i>PURCH_ITEM_CMPLT_BYTE_STREAM</i> ).....	41
4.2.11	Purchase Complete Byte Stream ( <i>PURCH_CMPLT_BYTE_STREAM</i> ).....	42
4.2.12	Game Play Byte Stream ( <i>GAME_PLAY_BYTE_STREAM</i> ) .....	42
4.2.13	Game Result Byte Stream ( <i>GAME_RESULT_BYTE_STREAM</i> ) .....	43
4.2.14	Progressive Payout Byte Stream ( <i>PROGRESSIVE_BYTE_STREAM</i> ) .....	44
4.2.15	Game Statistics Byte Stream ( <i>GAME_STATS_BYTE_STREAM</i> ) .....	44
4.2.16	Game Station Statistics Byte Stream ( <i>GS_STATS_BYTE_STREAM</i> ).....	45
4.2.17	Status Byte Stream ( <i>STATUS_BYTE_STREAM</i> ) .....	46
4.2.18	Acknowledge Message Byte Stream ( <i>ACKMSG_BYTE_STREAM</i> ).....	46
4.2.19	Acknowledged Message Type Byte Stream ( <i>ACKMT_BYTE_STREAM</i> ) .....	47
4.2.20	Win Level Configuration Byte Stream ( <i>WLCONFIG_BYTE_STREAM</i> ).....	48
4.2.21	Game Configuration Byte Stream ( <i>GMCONFIG_BYTE_STREAM</i> ) .....	48
4.2.22	Game Station Configuration Byte Stream ( <i>GSCONFIG_BYTE_STREAM</i> ).....	49
4.2.23	ATM Request Byte Stream ( <i>ATM_REQUEST_BYTE_STREAM</i> ) .....	50
4.2.24	Player Rating Session Byte Stream ( <i>PRATS_SESSION_BYTE_STREAM</i> ) .....	50
4.2.25	Game Play Session Byte Stream ( <i>PRATS_GAME_PLAY_BYTE_STREAM</i> ).....	51

---

4.2.26	Wager Detail Byte Stream ( <i>WAGER_DETAIL_BYTE_STREAM</i> ) .....	52
4.2.27	Originating Message Byte Stream ( <i>ORIG_MSG_BYTE_STREAM</i> ) .....	52
4.2.28	Draw Event Byte Stream ( <i>DRAW_EVENT_BYTE_STREAM</i> ) .....	53
4.2.29	Draw Result Byte Stream ( <i>DRAW_RESULT_BYTE_STREAM</i> ) .....	53
<b>5</b>	<b>OGP FIELD CODE DEFINITIONS .....</b>	<b>54</b>
<b>5.1</b>	<b>Summary of Message Types .....</b>	<b>54</b>
<b>5.2</b>	<b>Account Number Type Definitions .....</b>	<b>57</b>
<b>5.3</b>	<b>Status Code Definitions .....</b>	<b>58</b>
<b>5.4</b>	<b>Message Flag Definitions .....</b>	<b>59</b>
<b>5.5</b>	<b>Error Code Definitions .....</b>	<b>60</b>
<b>5.6</b>	<b>User Information Flag Definitions .....</b>	<b>63</b>
<b>5.7</b>	<b>Game Type Definitions .....</b>	<b>63</b>
<b>5.8</b>	<b>ATM Transaction Code Definitions .....</b>	<b>64</b>
<b>5.9</b>	<b>Player Rating Category Code Definitions .....</b>	<b>64</b>
<b>6</b>	<b>OGP COMMUNICATION REQUIREMENTS .....</b>	<b>65</b>
<b>6.1</b>	<b>RS-232 Serial Communications Protocol .....</b>	<b>65</b>
<b>6.2</b>	<b>TCP/IP Communications Protocol .....</b>	<b>65</b>
<b>6.3</b>	<b>Game Type Specific Protocols .....</b>	<b>66</b>
6.3.1	Purchase with Play .....	66
6.3.2	Purchase without Play .....	66
6.3.3	Play without Purchase .....	67
<b>6.4</b>	<b>Acknowledge Protocol .....</b>	<b>67</b>
<b>APPENDIX A - OGP MESSAGE SEQUENCE DIAGRAMS .....</b>		<b>68</b>
<b>APPENDIX B - OGP C/C++ LANGUAGE COMPATIBLE DEFINITIONS .....</b>		<b>75</b>

## List of Tables and Figures

<b>Figure 1 – Open Gaming Protocol Context Diagram .....</b>	<b>6</b>
<b>Table 5.1.1 – Game Station to Transaction Processing System Messages .....</b>	<b>54</b>
<b>Table 5.1.2 – Transaction Processing System to Game Station Messages .....</b>	<b>55</b>
<b>Table 5.2 – Account Number Types .....</b>	<b>57</b>
<b>Table 5.3 – Game Station Status Codes .....</b>	<b>58</b>
<b>Table 5.4 – Message Flags .....</b>	<b>59</b>
<b>Table 5.5 – OGP Error Codes .....</b>	<b>60</b>

**Table 5.6 – User Information Flags.....63**  
**Table 5.7 – Game Types .....63**  
**Table 5.8 – ATM Transaction Type Codes.....64**  
**Table 5.9 – Player Rating Category Type Codes .....64**

## **1 SCOPE**

### **1.1 Identification**

This document specifies the Open Gaming Protocol (OGP) for the Standing Stone Gaming, Inc. cashless casino gaming system. The owner and governing technical organization for the OGP is Standing Stone Gaming, Inc. (SSG, Inc.), Verona, New York. The points of contact for all matters related to the OGP are Mr. Jim Lavoie and Mr. Edward Hole of Rite-Solutions, Inc. The document provides an overview of the protocol, applicable context, protocol definition, and implementation guidance.

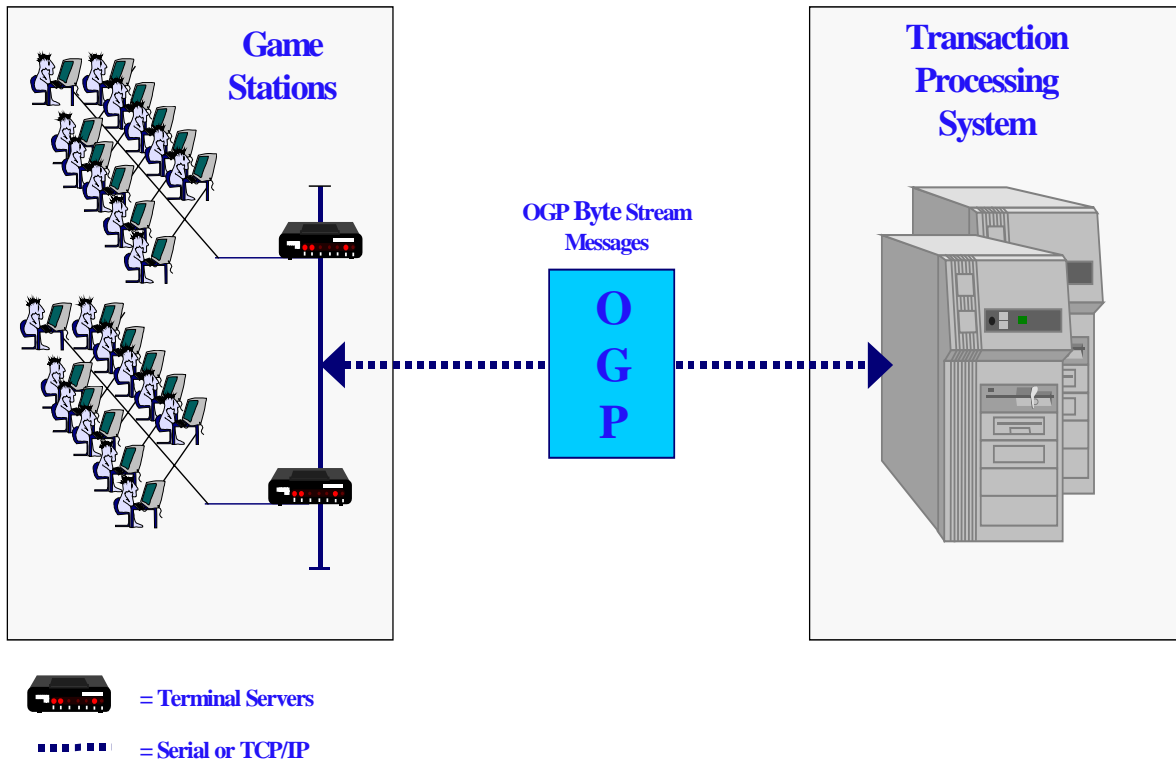
### **1.2 Overview**

OGP is a byte stream protocol developed for use in cashless transaction gaming systems. The protocol is intended to be vendor neutral and facilitate the development of cashless casino gaming systems capable of supporting multiple vendor game station environments. The protocol defines the set of messages passed between a game terminal and a cashless casino transaction processing system. The protocol also defines the communication environment. Figure 1 describes the gaming system context and interface supported by the OGP. OGP messages are transmitted as a series of bytes over a serial or network connection. Each message is defined to be a sequence of one or more byte streams, where each byte stream is composed of byte stream specific fields. OGP is based on the protocol currently in-use by the Turning Stone Casino supporting their cashless casino gaming system. The protocol can be implemented over RS-232, RS-422, or TCP/IP based connections. Note: RS-485 or other multi-drop serial networks can be accommodated with the SSG-MDA.

### **1.3 Compliance**

OGP is intended to be an open (i.e. public) and extensible protocol implemented over industry standard communication architectures. This specification shall be made available to the gaming vendor public to allow implementation of the protocol in their environment. To facilitate use of the OGP, extensions and/or changes to the protocol will be documented and distributed in this

specification or subsequent revisions of this specification. Requests for the latest version of the OPG Specification or requests for additions or changes to the OGP should be directed to Mr. Jim Lavoie or Mr. Edward Hole at Rite-Solutions, Inc, Middletown, RI, (401) 847-3399. A copy of this OGP specification can be downloaded at <http://standingstonegaming.com/docs/ipd-ogp.pdf> in PDF format.



**Figure 1 – Open Gaming Protocol Context Diagram**

**1.4 Document Organization**

This document is organized as follows:

- Section 1 This section describes the nature and scope of the document including identification, overview of the OGP, compliance, and document organization information.
- Section 2 This section describes the conventions used by the specification to define and describe the OGP.
- Section 3 This section describes OGP message formats including the message name, symbolic name, description, message ID, type, and message response requirements.
- Section 4 This section provides the byte stream and byte stream field definitions.
- Section 5 This section provides predefined field code values and definitions including status codes, acknowledge codes, flags, and miscellaneous code values and definitions.
- Section 6 This section provides communication environment requirements.
- Appendix A This appendix provides OGP message sequence diagrams showing generalized OGP message flow over time.
- Appendix B This appendix provides the C/C++ language compatible definitions for OGP in header file format.

## **2 OGP BASICS AND DOCUMENT CONVENTIONS**

The following sections outline the document conventions, notations and other basic information required to define and describe the Open Gaming Protocol.

### **2.1 Data Types and Sizes**

#### **2.1.1 Notation**

The following notation is used to indicate data type and size information.

**B** Specifies a 1-byte unsigned integer value.

**C** Specifies a 1-byte character from the ISO-Latin 1 character set.

**I2** Specifies a 2-byte signed integer value.

**U2** Specifies a 2-byte unsigned integer value.

**I4** Specifies a 4-byte signed integer value.

**U4** Specifies a 4-byte unsigned integer value.

**<n>** Follows a data type or byte stream type and denotes a sequence of *n* values of that type.

#### **2.1.2 Integer Values**

All byte stream integer values are to be transmitted in *network byte order*. Network byte order, also referred to as *big endian* byte order, assumes that the most significant byte of the integer value is stored in the lowest memory address.

#### **2.1.3 Character Sequences (Strings)**

All character sequences (specified as **C<n>**) are to be interpreted as null-terminated character strings unless otherwise specified. A null or zero (0) byte is used to mark the end of the string and is required. This implies that the maximum length of the string is (*n-1*) characters. Although strings shorter than the sequence size are permitted, *n* bytes of

data must be transmitted as a part of a byte stream. All bytes following the string termination byte, up to the end of the character sequence, must also be null bytes.

## 2.2 Font Conventions

The following font conventions are used throughout this document:

<b>FIXED_BOLD</b>	Emboldened, fixed width fonts are used for symbolic message names and data type specifications.
<b><i>FIXED_ITALICS</i></b>	Bold, italicized fixed width fonts are used as symbolic names for pre-defined byte streams.
<b>FIXED_WIDTH</b>	Fixed width fonts in upper case are used for symbolic names and constant values. These values are defined in the OGP .h header file provided in <a href="#">Appendix B</a> .
<i>italics</i>	Fixed width italic fonts and lower case are used to specify byte stream field names or any other parameters which are to be replaced with a user provided value.
<i>italics</i>	All other italicized text is used for emphasis or the introduction of a new term or concept.

## 2.3 Message Definitions

Each message definition in this document is presented in the following format:

**Message Name (SYMBOLIC\_MESSAGE\_NAME)**

Description:	A textual description of the message including its purpose, the general content of the message, solicitation and response information.
Message ID:	The integer message identifier value (in decimal).
Type:	Indicates whether the message is solicited or unsolicited by the receiver. Some messages can be both solicited and unsolicited.
Format:	Names and order of the byte stream definitions comprising the message content. Note, that byte stream names may be followed by

the sequence operator **<n>**, indicating that there are *n*-instances of the specified byte stream.

Response: A textual description of the required response to this message and any other required actions upon receipt of this message type.

**Example:**

**3.2.1 Game Station Status Message (OGP\_GSTP\_STATUS)**

Description: A Game Station sends terminal level status information to the transaction processing system in response to a Status Request message (**OGP\_TPGS\_STATUSREQ**). The Game Station can also send a Game Station Status message at any time to notify the transaction processing system of a change in Game Station status. If the status message is being sent in response to an **OGP\_TPGS\_STATUSREQ** message, the Game Station must set the value of the *serial\_no* field in the **MSG\_HEADER\_BYTE\_STREAM** to the value of the *serial\_no* field received in the **OGP\_TPGS\_STATUSREQ** message.

Message ID: 1001

Type: Solicited, Unsolicited.

Format: **START\_MSG\_BYTE\_STREAM**  
**MSG\_HEADER\_BYTE\_STREAM**  
**STATUS\_BYTE\_STREAM**  
**END\_MSG\_BYTE\_STREAM**

Response: The transaction processing system will send an Acknowledge message (**OGP\_TPGS\_ACK**) in response to the Game Station Status message.

## 3 OGP MESSAGE FORMATS

### 3.1 General Message Format

All OGP messages are composed of 5 mandatory parts: 1) a start of message word, 2) a message size word, 3) the message contents, 4) a checksum word, and 5) an end of message word. OGP messages have the following general format:

```
[ start_of_msg_word U4 ]  
[ message_size U4 ]  
[ message_contents B<message_size> ]  
[ checksum_word U4 ]  
[ end_of_msg_word U4 ]
```

#### 3.1.1 Start of Message Word

Each OGP message starts with a 4-byte word that indicates the beginning of the message. This word is a message framing word that provides a special 4-byte pattern that indicates the start of the message. The 4-byte pattern is defined in the OGP Header File in Appendix B (OGP\_START\_FRAME\_WORD). The frame word can be used to resynchronize message processing software in the event that a corrupted byte stream is encountered.

#### 3.1.2 Message Size Word

A 4-byte unsigned integer value specifying the message size (in bytes) follows the Start of Message Word. This value represents the size of the DES encrypted message contents part. The message size value is used to read the message contents, which immediately follows in the byte stream. An OGP message may contain at most OGP\_MAX\_MSG\_SIZE bytes (currently 1024 bytes), including the start of message, message size, checksum and end of message words. An application program implementing this protocol may take advantage of this fact by reading data into a static buffer versus dynamically allocating a data buffer for each message.

### **3.1.3 Message Contents**

The message contents part represents the body of the message and is composed of one or more byte streams of data. The number of bytes contained in the message contents part is specified in the message size word. Unless specifically waived by the Jurisdiction, the OGP message contents part shall be DES encrypted. Note that the sizes of the encrypted and decrypted message contents part may be different.

### **3.1.4 Message Checksum Word**

The message checksum word follows the message contents. This 4-byte unsigned integer value, a 32-bit checksum Cyclic Redundancy Check (CRC), is the CRC value computed on each byte of the unencrypted message contents (i.e. not including the message size, checksum, and frame words). The CRC is computed per the POSIX 1003.2 standard for computing 32-bit CRCs.

### **3.1.5 End of Message Word**

The End of Message Word immediately follows the checksum word and provides a special 4-byte pattern that indicates the end of the message. The 4-byte pattern is defined in the OGP Header File in [Appendix B](#) (OGP\_END\_FRAME\_WORD). This word can be used to resynchronize the message processing software in the event that a corrupted byte stream is encountered. The word immediately following the message frame word should be the beginning of a new message stream.

## **3.2 Game Station Originated Messages**

The following subsections provide detailed descriptions of OGP messages that originate at a game station and are sent to the transaction processing system.

### 3.2.1 Game Station Boot Message (OGP\_GSTP\_GSBOOT)

- Description: A game station sends a Game Station Boot Message to the transaction processing system to request an initial game terminal configuration. The **OGP\_GSTP\_GSBOOT** message is the very first message that a game station sends to the transaction processing system. The Game Station Boot Message contents part consists solely of the standard message header stream.
- Message ID: 1000
- Type: Unsolicited
- Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)
- Response: The transaction processing system will send a Game Station Configuration Message ([OGP\\_TPGS\\_GSCONFIG](#)) in response to a Game Station Boot Message, if the boot request is being made from a known source. If a boot request is received from an unknown game terminal or an unknown IP address, the transaction processing system will send an Acknowledge Message ([OGP\\_TPGS\\_ACK](#)) with the `OGP_ACK_NEG` flag set and the *reason* field set to indicate why the boot request failed.

### 3.2.2 Game Station Status Message (OGP\_GSTP\_STATUS)

- Description: A Game Station sends terminal status information to the transaction processing system in response to a Status Request Message ([OGP\\_TPGS\\_STATUSREQ](#)). The Game Station can also send a Game Station Status Message at any time, to notify the transaction processing system of a change in Game Station status.

If the Game Station Status Message is being sent in response to a Status Request Message, the Game Station must set the value of the *serial\_no* field in the [MSG\\_HEADER\\_BYTE\\_STREAM](#) to the

value of the *serial\_no* field in the [OGP\\_TPGS\\_STATUSREQ](#) message received by the Game Station.

The Acknowledge byte stream contained in the Status Message can be used by the Game Station to provide amplifying information for the status code being sent as part of the message. The Acknowledge byte stream provides for a major and minor reason code, in addition to providing for an explanatory text message. If the Game Station does not provide any amplifying information, the Acknowledge byte stream should be sent as all null bytes.

Message ID: 1001

Type: Solicited, Unsolicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[STATUS\\_BYTE\\_STREAM](#)  
[ACKMSG\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Response: The transaction processing system will send an Acknowledge Message ([OGP\\_TPGS\\_ACK](#)) in response to a Game Station Status Message.

### **3.2.3 Player Logon Request Message (OGP\_GSTP\_LOGON)**

Description: The Game Station sends a Player Logon Request Message to the transaction processing system, when a player inserts an account card into the Game Station card reader. This message is used to initiate a player session at a game station. Entry of an account holder PIN is typically required, but is dependent upon the type of player card that is inserted into the reader.

Message ID: 1002

Type: Unsolicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)

MSG\_HEADER\_BYTE\_STREAM

ACCTNO\_BYTE\_STREAM

END\_MSG\_BYTE\_STREAM

Response: If the player logon is successful, the transaction processing system will send a Player Logon Response Message (OGP\_TPGS\_LOGON) containing the player's current account balance, user account information, preferences, etc. If the player logon sequence cannot be completed, the transaction processing system will send an Acknowledge Message (OGP\_TPGS\_ACK) with the OGP\_ACK\_NEG flag set and the *reason* field set to indicate why the logon failed.

### 3.2.4 Player Logoff Request Message (OGP\_GSTP\_LOGOFF)

Description: The Player Logoff Request Message terminates a player's session at the game station. The game station can send a Player Logoff Request Message to the transaction processing system at any time (typically, when the account card is pulled from the card reader).

Message ID: 1003

Type: Unsolicited

Format: START\_MSG\_BYTE\_STREAM  
MSG\_HEADER\_BYTE\_STREAM  
ACCTNO\_BYTE\_STREAM  
END\_MSG\_BYTE\_STREAM

Response: The transaction processing system responds with a Player Logoff Response Message (OGP\_TPGS\_LOGOFF) to indicate successful logoff of the player from the system. The Player Logoff Response Message returns the player's final account balance, which should be verified against the balance maintained by the game station. If any discrepancy exists, game play at the station should be halted and the attendant call light should be lit.

In the event that the player logoff sequence cannot be completed, the transaction processing system will send an Acknowledge Message

([OGP\\_TPGS\\_ACK](#)) with the OGP\_ACK\_NEG flag set and the *reason* field set to indicate why the logoff failed.

### 3.2.5 Game Purchase Request Message (OGP\_GSTP\_PURCHREQ)

**Description:** The Game Purchase Request Message requests one or more chances, plays, or tickets from the transaction processing system. The game station can send a Game Purchase Request Message to the transaction processing system at any time after a successful player logon and prior to a player logoff. Purchases are specified on a per game basis and multiple chances, plays, or tickets may be purchased using a single purchase request.

**Message ID:** 1004

**Type:** Unsolicited

**Format:** [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[ACCTNO\\_BYTE\\_STREAM](#)  
[PURCH\\_REQ\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

**Responses:** If the Game Purchase Request can be filled, the transaction processing system responds with a Purchase Message ([OGP\\_TPGS\\_PURCH](#)). If the Purchase Request cannot be filled, the transaction processing system will send an Acknowledge Message ([OGP\\_TPGS\\_ACK](#)) with the OGP\_ACK\_NEG flag set and the *reason* field set to indicate why the order could not be filled.

### 3.2.6 Game Play Message (OGP\_GSTP\_GAME\_PLAY)

**Description:** The game station sends a Game Play Message to the transaction processing system to indicate that a game play, game chance, or game ticket reveal has taken place. The amount of the win (credit) or loss (debit) is provided as part of the message stream. The game station can send a Game Play Message to the transaction processing system at any time after a successful player logon and prior to a player logoff.

The Game Play message is used for **purchase-with-play** type games and **play-without-purchase** type games.

For **purchase-with-play** type games, a Game Purchase Request Message ([OGP\\_GSTP\\_PURCHREQ](#)) is used to purchase the game play, game chance, or ticket. Following the purchase, a Game Play Message is then sent to reveal the result of the win or loss. The *serial\_no* field in the [MSG\\_HEADER\\_BYTE\\_STREAM](#) must match the *serial\_no* field of the preceding Game Purchase Request Message. A loss is represented as zero (0) in the *amount* field of the [GAME\\_PLAY\\_BYTE\\_STREAM](#).

For **play-without-purchase** type games, there is no prior Purchase Request and a loss is represented as a negative number in the *amount* field of the [GAME\\_PLAY\\_BYTE\\_STREAM](#).

See section [6.3 Game Type Specific Protocols](#) for additional information on the various game types. See also [Appendix A](#) for sequence diagrams that illustrate the OGP protocol differences between the various game types.

Message ID: 1005

Type: Unsolicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[ACCTNO\\_BYTE\\_STREAM](#)  
[GAME\\_PLAY\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Responses: The transaction processing system responds with a Game Result Message ([OGP\\_TPGS\\_GAME\\_RESULT](#)) if the gaming transaction is successfully processed. If the gaming transaction cannot be recorded, the transaction processing system will send an Acknowledge Message ([OGP\\_TPGS\\_ACK](#)) with the OGP\_ACK\_NEG flag set and the *reason* field set to indicate why the game play transaction could not be completed. If an account or transaction related negative acknowledgement is received, the game station should instruct the player to contact Customer Service personnel.

### 3.2.7 User Preferences Message (OGP\_GSTP\_USERPREF)

Description: The Game Station can send a User Preferences Message to the transaction processing system at any time in order to update the user's gaming preferences information maintained in the user's account.

Message ID: 1006

Type: Unsolicited

Format: START\_MSG\_BYTE\_STREAM  
MSG\_HEADER\_BYTE\_STREAM  
ACCTNO\_BYTE\_STREAM  
USER\_PREF\_BYTE\_STREAM  
END\_MSG\_BYTE\_STREAM

Responses: The transaction processing system responds to the User Preferences Message with an Acknowledge Message (OGP\_TPGS\_ACK) to indicate the success (OGP\_ACK\_POS) or failure (OGP\_ACK\_NEG) of the user preferences update transaction. In the event of a negative acknowledgement, the *reason* field will be set to an error code indicating why the user preferences could not be updated.

### 3.2.8 Game Station Statistics Message (OGP\_GSTP\_GS\_STATS)

Description: A Game Station sends a Game Station Statistics Message to the transaction processing system in response to a Game Station Statistics Request Message (OGP\_TPGS\_GS\_STATSREQ). The Game Station Statistics Message provides game station specific game statistics and kiosk meter readout information to the transaction processing system.

The game station must set the value of the *serial\_no* field in the MSG\_HEADER\_BYTE\_STREAM to the value of the *serial\_no* field received in the OGP\_TPGS\_GS\_STATSREQ message.

Message ID: 1007

Type: Solicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[GS\\_STATS\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Response: The transaction processing system will send an Acknowledge message ([OGP\\_TPGS\\_ACK](#)) in response to the Game Station Statistics message.

### 3.2.9 Acknowledge Message (OGP\_GSTP\_ACK)

Description: An Acknowledge Message is sent to the transaction processing system under the following two circumstances: 1) for any messages not explicitly acknowledged by other game station messages, and 2) whenever an expected explicit acknowledge message cannot be sent due to a failure in the processing of a request message. In the first case, the [OGP\\_GSTP\\_ACK](#) message is sent indicating the success ([OGP\\_ACK\\_POS](#) message flag set) or failure ([OGP\\_ACK\\_NEG](#) message flag set) of the transaction. In the second case, the message *flags* field should have the [OGP\\_ACK\\_NEG](#) bit set.

All negatively acknowledged messages must set the *reason* field of the [ACKMSG\\_BYTE\\_STREAM](#) to an appropriate error code to indicate the reason for the negative acknowledgement. See section [5.5 Error Code Definitions](#) for the list of error codes that can be specified.

Note that the Game Station must always set the *serial\_no* field in the [MSG\\_HEADER\\_BYTE\\_STREAM](#) to the value of the *serial\_no* field in the transaction processing system message that is being acknowledged.

The [ORIG\\_MSG\\_BYTE\\_STREAM](#) is an optional byte stream, and if provided, identifies the message content that is being acknowledged. The [OGP\\_INCL\\_ACKD\\_MSG](#) flag should be set in the *flags* field of the [MSG\\_HEADER\\_BYTE\\_STREAM](#) if this message byte stream is included.

See also section [6.4 Acknowledge Protocol](#) for additional information on message acknowledgement within the Open Gaming Protocol.

Message ID: 1008

Type: Solicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[ACKMSG\\_BYTE\\_STREAM](#)  
[ACKMT\\_BYTE\\_STREAM](#)  
[ORIG\\_MSG\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Responses: None.

### **3.2.10 ATM Request Message (OGP\_GSTP\_ATM\_REQUEST)**

Description: The ATM Request Message provides the capability to perform financial transactions on a player account. This message can be used to deposit money into a player account, withdraw money from a player account or transfer money to/from another account owned by the player. The ATM Request Message provides the means for money to be deposited into a player's account from a Game Terminal (e.g. through the use of a bill acceptor) or through kiosks placed throughout the casino that allow a player to transfer funds electronically.

Message ID: 1009

Type: Unsolicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[ATM\\_REQUEST\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Responses: The transaction processing system will send an Account Update Message ([OGP\\_TPGS\\_ACCT\\_UPDATE](#)) in response to the ATM Request Message, which will provide the updated player account balance. If the ATM transaction cannot be executed, the transaction processing system will send an Acknowledge Message ([OGP\\_TPGS\\_ACK](#)) with the OGP\_ACK\_NEG flag set and the *reason* field set to indicate why the ATM transaction could not be completed.

### 3.2.11 Player Rating Message (OGP\_GSTP\_RATING)

Description: The Game Station sends a Player Rating Message to the transaction processing system when a player removes his or her account card from the game station or performs a player logoff operation terminating the gaming session. This message is used to provide summary play data to the transaction processing system for player rating point calculation.

Message ID: **1010**

Type: Unsolicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[PRATS\\_SESSION\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Response: The transaction processing system will send an Acknowledge Message ([OGP\\_TPGS\\_ACK](#)) in response to a Player Rating Message indicating the success (OGP\_ACK\_POS message flag bit set) or failure (OGP\_ACK\_NEG message flag bit set) of the transaction.

## 3.3 Transaction Processing System Originated Messages

The messages defined in the following subsections are sent from the Transaction Processing System to a Game Station.

### 3.3.1 Status Request Message (OGP\_TPGS\_STATUSREQ)

- Description: The transaction processing system can send a Status Request Message to the Game Terminal at any time, in order to obtain the current status of the Game Terminal. This message requires only the standard header as its message content.
- Message ID: 1
- Type: Unsolicited
- Format: START\_MSG\_BYTE\_STREAM  
MSG\_HEADER\_BYTE\_STREAM  
END\_MSG\_BYTE\_STREAM
- Responses: The Game Station responds to a Status Request Message with a Game Station Status message (OGP\_GSTP\_STATUS).

### 3.3.2 Player Logoff Message (OGP\_TPGS\_LOGOFF)

- Description: The transaction processing system sends a Player Logoff Message to the Game Station in response to a Game Station Player Logoff Request Message (OGP\_GSTP\_LOGOFF). The transaction processing system can also send a Player Logoff Message to the Game Station at any time, to forcibly halt play at a Game Station for the current player.

If the *acctno\_type* field of the ACCTNO\_BYTE\_STREAM is set to OGP\_ACCT\_NO\_NONE in an unsolicited Player Logoff Message, Game Station should logoff any player that is currently logged onto the Game Station. Any other valid value for this field indicates that the player specified in the *acctno* field should be logged off of the Game Station. The OGP\_MSG\_UN SOLICITED flag is set by the Transaction Processor to differentiate a forced logoff (unsolicited) message from a logoff message solicited by a Game Station Player Logoff Request.

- Message ID: 2

Type: Unsolicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[ACCT\\_UPDT\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Responses: The Game Station responds with an Acknowledge Message ([OGP\\_GSTP\\_ACK](#)) to indicate the success (OGP\_ACK\_POS) or failure (OGP\_ACK\_NEG) of the player sign-off transaction. In the event of a negative acknowledgement, the *reason* field will be set to an error code indicating why the player sign-off could not be completed. In either case, the transaction processing system will not accept any further transactions from the player.

### 3.3.3 Purchase Message (OGP\_TPGS\_PURCH)

Description: The transaction processing system sends a Purchase Message to the Game Station in response to a Purchase Request message ([OGP\\_GSTP\\_PURCHREQ](#)), if the order can be filled. The Purchase Message explicitly acknowledges the Purchase Request Message by echoing back purchase information (the game identifier, number of tickets, total cost of the transaction) and returns the user's updated account balance after the purchase along with the requested game chances.

The *serial\_no* field in the [MSG\\_HEADER\\_BYTE\\_STREAM](#) of the OGP\_TPGS\_PURCH message must be set to the value of the *serial\_no* field contained in the [OGP\\_GSTP\\_PURCHREQ](#) message sent by the Game Station initiating the purchase.

Message ID: 3

Type: Solicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[PURCH\\_CMPLT\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Responses: The Game Station responds to a Purchase Message with an Acknowledge Message ([OGP\\_GSTP\\_ACK](#)) to indicate the success (OGP\_ACK\_POS message flag bit set) or failure (OGP\_ACK\_NEG message flag bit set) of the Purchase Message processing.

### 3.3.4 Player Logon Response Message (OGP\_TPGS\_LOGON)

Description: The transaction processing system sends a Player Logon Response Message to the Game Station in response to a Player Logon Request Message ([OGP\\_GSTP\\_LOGON](#)) received from the Game Station. This message positively acknowledges the Player Logon Request Message and provides full player account information, including current account balance and user preferences.

The *serial\_no* field in the [MSG\\_HEADER\\_BYTE\\_STREAM](#) of the **OGP\_TPGS\_LOGON** message must be set to the value of the *serial\_no* field contained in the [OGP\\_GSTP\\_LOGON](#) message sent by the Game Station.

Message ID: 4

Type: Solicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[ACCT\\_UPDT\\_BYTE\\_STREAM](#)  
[ACCT\\_INFO\\_BYTE\\_STREAM](#)  
[USER\\_PREF\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Responses: The Game Station responds with an Acknowledge Message ([OGP\\_GSTP\\_ACK](#)) to indicate the success (OGP\_ACK\_POS) or failure (OGP\_ACK\_NEG) of the logon sequence completion. In the event of a negative acknowledgement, the *reason* field will be set to an error code indicating why the logon sequence could not be completed.

### 3.3.5 Game Result Message (OGP\_TPGS\_GAME\_RESULT)

Description: The transaction processing system sends a Game Result Message to the Game Station to indicate the result of the preceding game play, chance, or ticket reveal received from the Game Station. The Game Result Message positively acknowledges a Game Play Message ([OGP\\_GSTP\\_GAME\\_PLAY](#)) and provides the user's updated account balance, the original game play data, and any game result data.

The *serial\_no* field in the [MSG\\_HEADER\\_BYTE\\_STREAM](#) of the **OGP\_TPGS\_GAME\_RESULT** message must be set to the value of the *serial\_no* field contained in the [OGP\\_GSTP\\_GAME\\_PLAY](#) message sent by the Game Station.

Message ID: 5

Type: Solicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[ACCT\\_UPDT\\_BYTE\\_STREAM](#)  
[GAME\\_RESULT\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Responses: The Game Station responds with an Acknowledge message ([OGP\\_GSTP\\_ACK](#)) to indicate the success (OGP\_ACK\_POS) or failure (OGP\_ACK\_NEG) of the account update. In the event of a negative acknowledgement, the *reason* field will be set to an error code indicating why the account update could not be completed.

### 3.3.6 Game Station Statistics Request Message (OGP\_TPGS\_GS\_STATSREQ)

Description: The transaction processing system can send a Game Station Statistics Request Message to the Game Station at any time to obtain the current game statistics and kiosk meter readings.

Message ID: 6

Type: Unsolicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Responses: The Game Station responds to this message with a Game Station Statistics Message ([OGP\\_GSTP\\_GS\\_STATS](#)).

### 3.3.7 Acknowledge Message (OGP\_TPGS\_ACK)

Description: An Acknowledge Message is sent to the game station under the following two circumstances: 1) for any messages not explicitly acknowledged by other transaction processing system messages, and 2) whenever an expected explicit acknowledge cannot be sent due to a failure in the processing of a request message. In the first case, the **OGP\_TPGS\_ACK** message is sent indicating the success (OGP\_ACK\_POS message flag set) or failure (OGP\_ACK\_NEG message flag set) of the transaction. In the second case, the message *flags* field should have the OGP\_ACK\_NEG bit set.

All negatively acknowledged messages must set the *reason* field of the [ACKMSG\\_BYTE\\_STREAM](#) to an appropriate error code to indicate the reason for the negative acknowledgement. See section [5.5 Error Code Definitions](#) for the list of error codes that can be specified.

Note that the transaction processing system must always set the *serial\_no* field in the [MSG\\_HEADER\\_BYTE\\_STREAM](#) to the value of the *serial\_no* field in the game station message that is being acknowledged.

The [ORIG\\_MSG\\_BYTE\\_STREAM](#) is an optional byte stream, and if provided, identifies the message content that is being acknowledged. The **OGP\_INCL\_ACKD\_MSG** flag should be set in the *flags* field of the [MSG\\_HEADER\\_BYTE\\_STREAM](#) if this message byte stream is included.

See also section [6.4 Acknowledge Protocol](#) for additional information on message acknowledgement within the Open Gaming Protocol.

Message ID: 7

Type: Solicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[ACKMSG\\_BYTE\\_STREAM](#)  
[ACKMT\\_BYTE\\_STREAM](#)  
[ORIG\\_MSG\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Responses: None.

### 3.3.8 Game Station Enable Message (OGP\_TPGS\_ENABLE)

Description: The transaction processing system sends a Game Station Enable Message to the Game Station to re-enable play on that station. The Enable Message contents part consists solely of the standard message header stream.

Message ID: 8

Type: Unsolicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Responses: The Game Station responds with an Acknowledge Message ([OGP\\_GSTP\\_ACK](#)) to indicate the success (OGP\_ACK\_POS) or failure (OGP\_ACK\_NEG) of the Game Station enable command. In the event of a negative acknowledgement, the *reason* field of the [ACKMSG\\_BYTE\\_STREAM](#) should be set to an error code indicating why the game station could not be enabled.

The transaction processing system may send a Status Request Message ([OGP\\_TPGS\\_STATUSREQ](#)) in response to a negative acknowledgement, in order to acquire additional information as to why the Game Station cannot be enabled.

### 3.3.9 Game Station Disable Message ([OGP\\_TPGS\\_DISABLE](#))

**Description:** The transaction processing system sends a Game Station Disable Message to the Game Station to suspend play on that station. The Disable Message contains an Acknowledge Message byte stream, which is used to provide an explanation to the Game Station as to why it is being disabled. The Game Station shall suspend all further play until it is re-enabled. Regardless of the acknowledgement status, the transaction processing system will cease to process orders from the disabled Game Station until it has been re-enabled.

**Message ID:** 9

**Type:** Unsolicited

**Format:** [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[ACKMSG\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

**Responses:** The Game Station responds with an Acknowledge Message ([OGP\\_GSTP\\_ACK](#)) to indicate the success ([OGP\\_ACK\\_POS](#)) or failure ([OGP\\_ACK\\_NEG](#)) of the Game Station Disable command. In the event of a negative acknowledgement, the *reason* field will be set to an error code indicating why the disable command could not be processed. The Transaction processing system may also send a Status Request message ([OGP\\_TPGS\\_STATUSREQ](#)) in response to a negative acknowledgement, in order to acquire additional information as to why the Game Station cannot be disabled.

### 3.3.10 Game Station Configuration Message ([OGP\\_TPGS\\_GSCONFIG](#))

**Description:** The transaction processing system sends a Game Station Configuration Message to a Game Station in order to establish certain operational parameters, such as which game sets to offer and the corresponding pay tables. The transaction processing system sends one or more Game Station Configuration Messages in response to a Game Station Boot Message ([OGP\\_GSTP\\_GSBOOT](#)) received from a game station. The transaction processing system can also send an unsolicited Game Station Configuration Message at any time, to reconfigure a game station on an as needed basis.

If the `OGP_MSG_INIT` message flag is set in the `flags` field of the [MSG\\_HEADER\\_BYTE\\_STREAM](#) for this message, the Game Station should initialize its play selections to those in the message, otherwise those play selections defined by the message should be added to the Game Station's current offerings.

**Message ID:** 10

**Type:** Solicited, Unsolicited

**Format:** [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[GSCONFIG\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

**Response:** The Game Station responds with an Acknowledge Message ([OGP\\_GSTP\\_ACK](#)) to indicate the success (`OGP_ACK_POS`) or failure (`OGP_ACK_NEG`) of game station configuration (or reconfiguration). In the event of a negative acknowledgement, the `reason` field will be set to an error code indicating why game station configuration failed.

**Note:** This message is likely to be modified in future versions of the Open Gaming Protocol. Possible future enhancements may include such features as screen definition information and/or software downloads to the Game Station.

### 3.3.11 Clock Synchronization Message ([OGP\\_TPGS\\_TIME](#))

Description: The transaction processing system can send a Clock Synchronization Message to the Game Station at any time, to force the Game Station to synchronize its system clock with the transaction processing system. The Clock Synchronization Message has no data, however, the Game Station should use the *time* field in the [MSG\\_HEADER\\_BYTE\\_STREAM](#) to set its clock. The transaction processing system uses the *time* field of the Acknowledge Message returned by the Game Station, to determine if the clock has been set correctly on the Game Station.

Message ID: 11

Type: Unsolicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Responses: The Game Station responds with an Acknowledge Message ([OGP\\_GSTP\\_ACK](#)) to indicate the success (OGP\_ACK\_POS) or failure (OGP\_ACK\_NEG) of clock synchronization. In the event of a negative acknowledgement, the *reason* field will be set to an error code indicating why the clock reset failed. If the Game Station responds with a negative acknowledgement or if the transaction processing system does not deem the clocks to be satisfactorily synchronized, the Game Station will be disabled.

### 3.3.12 Progressive Payout Message (OGP\_TPGS\_PROGRESSIVE)

Description: The transaction processing system may send a Progressive Payout Message to the Game Station at any time in order to inform the Game Station that a progressive payout for a specific game has been updated.

Message ID: 12

Type: Unsolicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)

MSG\_HEADER\_BYTE\_STREAM  
PROGRESSIVE\_BYTE\_STREAM  
END\_MSG\_BYTE\_STREAM

Responses: None (OGP\_ACK\_NREQ should be set). The Progressive Payout message is a broadcast message and should not be acknowledged.

Note: If the OGP\_MSG\_INIT flag is set in the *flags* field of the message header for this message, the progressive for the specified *game\_id* has been reset, otherwise the progressive is rolling.

### 3.3.13 Terminal Notify Message (OGP\_TPGS\_TERMNOTIFY)

Description: The transaction processing system may send a Terminal Notify Message to the Game Station at any time, to instruct the terminal to display the text portion of the message to the current player or on the terminal marquee.

If the OGP\_MSG\_BANNER flag is set in the *flags* field of the MSG\_HEADER\_BYTE\_STREAM, the message should be added to the current message displayed on the terminal marquee. If the OGP\_MSG\_BANNER and the OGP\_MSG\_INIT flags are set, the message should replace the current message displayed on the terminal marquee. If the OGP\_MSG\_BANNER flag is not set, the message should be displayed to the player currently logged onto the game station.

Message ID: 13

Type: Unsolicited

Format: START\_MSG\_BYTE\_STREAM  
MSG\_HEADER\_BYTE\_STREAM  
ACKMSG\_BYTE\_STREAM  
END\_MSG\_BYTE\_STREAM

Responses: The Game Station responds with an Acknowledge message (OGP\_GSTP\_ACK) to indicate the success (OGP\_ACK\_POS) or

failure (OGP\_ACK\_NEG) of the Terminal Notify Message. If the OGP\_ACK\_NREQ flag in the message header *flags* field is set, a positive acknowledge message should not be sent to the transaction processing system. In the event of a negative acknowledgement, the *reason* field should be set to an error code indicating why the terminal notify command could not be processed.

### 3.3.14 Account Balance Update Message (OGP\_TPGS\_ACCT\_UPDATE)

- Description: The transaction processing system sends an Account Balance Update Message to a Game Station or an ATM kiosk, in response to an ATM Request Message. The Account Balance Update Message echoes the account number to which the ATM transaction was applied and provides the updated account balance.
- Message ID: 14
- Type: Solicited
- Format: START\_MSG\_BYTE\_STREAM  
MSG\_HEADER\_BYTE\_STREAM  
ACCT\_UPDT\_BYTE\_STREAM  
END\_MSG\_BYTE\_STREAM
- Responses: The Game Station or Kiosk responds with an Acknowledge Message (OGP\_GSTP\_ACK) to indicate the overall success (OGP\_ACK\_POS) or failure (OGP\_ACK\_NEG) of the account transaction.

### 3.3.15 Draw Service Message (OGP\_TPGS\_DRAWSERVICE)

Description: The Game Draw Server periodically broadcasts (via UDP) Draw Service Messages to the game stations. A game station can use these messages to determine the results of play-without-purchase type Game Play Messages.

The OGP Draw Service Message (**OGP\_TPGS\_DRAWSERVICE**) is a binary representation of the Game Server's Draw Server message that is repacked into an OGP message format. This message is

provided for game stations (through a special communications application) that are not TCP/IP capable and do not have more than one serial port available.

This message is **never** encrypted; even if other messages sent through the communications channel are encrypted. The game station is required to provide the necessary message processing logic to handle this caveat.

This message should **never** be acknowledged.

Message ID: 99

Type: Unsolicited

Format: [START\\_MSG\\_BYTE\\_STREAM](#)  
[MSG\\_HEADER\\_BYTE\\_STREAM](#)  
[DRAW\\_RESULT\\_BYTE\\_STREAM](#)  
[END\\_MSG\\_BYTE\\_STREAM](#)

Responses: None.

## 4 OGP BYTE STREAMS

### 4.1 Byte Stream Definition Format

The byte stream definitions outlined in [Section 4.2](#) below, provide the byte stream symbolic name, a textual description of the byte stream, byte stream field definitions (including field name, field size and type, and field order), and any other pertinent information relevant to the byte stream. Byte stream data content is to be ordered exactly as listed in the Field Definitions block.

Each byte stream definition presented in this document has the following format:

**Byte Stream Name (*SYMBOLIC\_BYTE\_STREAM\_NAME*)**

Description:        A textual description of the byte stream including its purpose, any variations in usage of the byte stream or restrictions.

Field Definitions:

[*field\_name* **TYPE**] Each Field Definitions section consists of one or more field definitions. Each field definition provides the symbolic name for the field and its corresponding data type. See [Section 2.1 – Data Types and Sizes](#) for an overview of the conventions used to represent various data types and for any special data type considerations. Each field definition also provides descriptive text that outlines the field usage and any special use cases or restrictions.

***BYTE\_STREAM\_DEF*** A byte stream may also be defined in terms of other pre-defined byte streams. A byte stream symbolic name may be placed in a Field Definitions section to denote that all of the fields of the pre-defined byte stream are to appear in sequence, relative to the placement of the byte stream symbolic name.

Notes: This section is optional and may be used to convey information that is not appropriate to any of the other sections of the byte stream definition.

## 4.2 OGP Byte Stream Definitions

The following sections define the byte streams of which OGP messages are composed.

### 4.2.1 Start of Message Byte Stream (*START\_MSG\_BYTE\_STREAM*)

Description: The Start of Message Byte Stream provides the start of message framing word and the total number of bytes of data content in the message. This byte stream sequence is common to all OGP messages and is the first byte stream sequence in the message.

Field Definitions:

[*sframe\_word* U4] The start of message framing word contains a special bit pattern that indicates the start of the message. The frame word is defined in the OGP Header file in [Appendix B](#) as OGP\_START\_FRAME\_WORD.

[*msg\_size* U4] The total number of bytes contained in the message content part of the message.

Notes: The *msg\_size* byte count does not include the *START\_MSG\_BYTE\_STREAM* and [END\\_MSG\\_BYTE\\_STREAM](#). byte streams.

### 4.2.2 Message Header Byte Stream (*MSG\_HEADER\_BYTE\_STREAM*)

Description: The Message Header Byte Stream begins the message content part and provides message data common to all OGP messages including protocol version information, a message type identifier, a terminal identifier, transaction time, and serial number fields. The message

header byte stream must be the first byte stream sequence following the start of message byte stream ([START\\_MSG\\_BYTE\\_STREAM](#)).

Field Definitions:

[ <i>protocol</i> U2]	Specifies the identifier for the message protocol being used. Normally, this field should be set to <code>PROTOCOL_OGP</code> , which is defined in the <code>OGP.h</code> header file in <a href="#">Appendix B</a> . However, this field may be set to other protocol identifier values, to indicate a gaming protocol that has been translated to OGP.
[ <i>major_ver</i> B]	Specifies the major version number of the protocol.
[ <i>minor_ver</i> B]	Specifies the minor version number of the protocol.
[ <i>msg_type</i> U4]	This field specifies a unique four-byte message type identifier that specifies the message content type. See section <a href="#">5.1 Summary of Message Types</a> for specific message identifier values.
[ <i>flags</i> U4]	This field specifies any message flags that modify or add additional meaning to the content of a message. See section <a href="#">5.4 Message Flag Definitions</a> for the set of message flags that may be provided.
[ <i>term_id</i> C<16>]	Specifies a unique character string identifying the game station terminal that is the sender or intended receiver of the message. The terminal ID must be terminated with a null (0) byte.
[ <i>time</i> I4]	Time that the original message was sent or transaction was performed. Expressed as seconds since 00:00:00 GMT January 1, 1970. Note that for messages sent with the <code>OGP_MSG_REPEAT</code> flag set, the time field does <u>not</u> reflect the sender's current time.
[ <i>serial_no</i> U4]	This field specifies a message serial number. The serial number must be unique for each message that is originated by a sender, or set to the value of the <i>serial_no</i> field of the message to which a response is being sent.
[ <i>trans_id</i> B<8>]	This field specifies an eight-byte value that uniquely identifies the transaction. This field is currently only used by the Transaction Processing System. For Game Station originated messages, this field should be set to all null

bytes. The Transaction Processor uses the first two bytes of this field to indicate the Transaction Processor instance that processed the transaction encapsulated by this message. The remaining six bytes are used as a 48-bit transaction counter. Non-transaction based messages sent by the Transaction Processor should also set this field to all null bytes.

Notes:                   None.

#### **4.2.3 End of Message Byte Stream (*END\_MSG\_BYTE\_STREAM*)**

Description:           The End of Message Byte Stream provides a message checksum value and end frame word. This byte stream is common to all OGP messages. It is the last byte stream in a message and immediately follows the message content part.

Field Definitions:

[*checksum* U4]           This 4-byte unsigned integer value, a 32-bit checksum Cyclic Redundancy Check (CRC), is the CRC value computed on each byte of the unencrypted message contents (i.e. not including the message size, checksum, and frame words). The CRC is computed per the POSIX 1003.2 standard for computing 32-bit CRCs.

[*eframe\_word* U4]       The end of message framing word contains a special bit pattern that indicates the end of the message and terminates all OGP messages. The frame word is defined in the OGP Header file in Appendix B as *OGP\_END\_FRAME\_WORD*.

Notes:                   The end of message byte stream is not included in the *msg\_size* byte count.

#### **4.2.4 Account Number Byte Stream (*ACCTNO\_BYTE\_STREAM*)**

Description:           The Account Number Byte Stream is used to pass user account information throughout the system. This byte stream contains the

user account number and Personal Identification Number (PIN) associated with a specific customer.

Field Definitions:

[ <i>acctno</i> C<32>]	This field specifies the account number to which the transaction is to be applied. The last byte of the account number must be a null (0) byte.
[ <i>PIN</i> C<16>]	The Personal Identification Number (PIN) entered by the account holder at game station logon, to access the account and allow purchases, credits, debits, and preference changes. This field should be set to all zeroes (i.e. null bytes), if a PIN is not required for the specified account number. The PIN is unencrypted.
[ <i>acctno_type</i> U4]	Account number type. See section <a href="#">5.2 Account Number Type Definitions</a> .

Notes:               None.

#### **4.2.5 Account Information Byte Stream (*ACCT\_INFO\_BYTE\_STREAM*)**

Description:        The Account Information Byte Stream is used to provide user account specific information. It includes the user's full name, address, date of birth, account balance, date of last visit, and last game played.

Field Definitions:

[ <i>full_name</i> C<32>]	Character string specifying the account holder's full name, as contained in the account. The string must be terminated with a null (0) byte.
[ <i>str_addr</i> C<32>]	This field is a character string specifying the street address of the account holder. The string must be terminated with a null (0) byte.
[ <i>city</i> C<24>]	Character string that specifies the account holder's city or town of residence. The string must be terminated with a null (0) byte.

[ <i>state</i> C<8>]	Character string specifying the account holder's state or province of residence. The string must be terminated with a null (0) byte.
[ <i>country</i> C<16>]	Character string specifying the account holder's country of residence. The string must be terminated with a null (0) byte.
[ <i>zip</i> C<16>]	Character string specifying the account holder's ZIP or Postal code. The string must be terminated with a null (0) byte.
[ <i>dob</i> C<12>]	Character string specifying the account holder's date of birth in the form MM-DD-YYYY, where MM = month of year (01-12), DD=day of month (01-31), YYYY = year (1900 – 2099). The string must be terminated with a null (0) byte.
[ <i>dolv</i> C<12>]	Character string specifying the date of the account holder's last visit to this casino. The string must be terminated with a null (0) byte.

Notes:               None.

#### **4.2.6 Account Update Byte Stream (*ACCT\_UPDT\_BYTE\_STREAM*)**

Description:        The Account Update Byte Stream is used to send updated account balance information to the game station.

Field Definitions:

***ACCTNO\_BYTE\_STREAM***

[ <i>balance</i> I4]	This field represents the account holder's current balance in pennies.
[ <i>session_id</i> U4]	A four-byte unsigned integer value that uniquely identifies a player session. The transaction processing system uses this field internally. This field has no meaning for Game Stations, however all bytes of the stream must be read.

Notes:               None.

#### 4.2.7 User Preferences Byte Stream (*USER\_PREF\_BYTE\_STREAM*)

Description: The User Preferences Byte Stream is used to send updated account holder gaming preferences to the game station or the transaction processing system. The byte stream includes gaming preferences (e.g. favorite games, typical time of play, machine type and location, symbol sets) and personal preferences (e.g. favorite drink, smoker/non-smoker).

##### Field Definitions:

[ <i>pflags</i> U4]	Preferences Flag. See section <a href="#">5.6 User Information Flag Definitions</a> .
[ <i>pref_name</i> C<24>]	Character string specifying the account holder's preferred name or nickname. The string must be terminated with a null (0) byte.
[ <i>fgame_id</i> C<16>]	This field represents the game ID of the account holder's favorite game.
[ <i>fsym_set</i> U4]	The game symbol set identifier preferred by the account holder.
[ <i>fmach_type</i> U4]	The type of gaming station preferred by the account holder.
[ <i>fmach_loc</i> U4]	The account holder's preferred location to play (e.g. bank, gaming room, non-smoking, etc.).
[ <i>fterm_ID</i> C<16>]	Character string specifying the terminal ID of the account holder's preferred game station
[ <i>ttime</i> U4]	Account holder's typical or preferred time of day to visit the casino gaming stations. Specified in military time: 0000-2359.
[ <i>tseason</i> U4]	The season at which the account holder prefers to visit the casino.
[ <i>fcocktail</i> C<32>]	The string identifier for the cocktail preferred by the account holder when visiting the casino.
[ <i>fsoft_drnk</i> C<32>]	The string identifier for the non-alcoholic soft drink preferred by the account holder when visiting the casino.
[ <i>fcigarette</i> C<32>]	The string identifier for the account holder's preferred cigarette brand.

[ <i>frest</i> <b>U4</b> ]	The identifier of the restaurant preferred by the account holder when visiting the casino.
[ <i>language1</i> <b>U2</b> ]	Preferred language.
[ <i>language2</i> <b>U2</b> ]	The account holder's second language choice if <i>language1</i> is not available.
[ <i>lucky_nums</i> <b>B&lt;64&gt;</b> ]	The player's set of "lucky numbers". The usage of this field may vary across transaction processing system installations, depending upon the games that are installed at the site. The default interpretation of this field is as follows:  [ <b>I4</b> ] The first four bytes are to be interpreted as a signed integer value and represent the player's lucky "play" or "draw" number.  [ <b>B&lt;8&gt;</b> ] The next eight bytes are to be interpreted as unsigned octet values in the range of 0-255 and represent the player's lucky lottery numbers.  [ <b>B&lt;20&gt;</b> ] The next 20 bytes are to be interpreted as unsigned octet values in the range of 0-255 and represent the player's lucky keno numbers.

Notes: None.

#### **4.2.8 Purchase Item Request Byte Stream (*PURCH\_ITEM\_REQ\_BYTE\_STREAM*)**

Description: The Purchase Item Request Byte Stream is used to describe the request to purchase a single item, chance, play, or ticket. It provides the game ID associated with the purchase, item type, item specific data, and cost of the purchase.

##### Field Definitions:

[ <i>game_id</i> <b>C&lt;16&gt;</b> ]	The <i>game_id</i> field specifies a unique game identifier, which indicates the type of game being played by the customer at the game station.
[ <i>item_type</i> <b>U4</b> ]	This field specifies a game specific item identifier, indicating the type of item, chance, play, or ticket being purchased.

[*item\_data* B<64>] Item specific data associated with the purchase.  
[*item\_cost* U4] The cost of the item, chance, play, or ticket in pennies.

Notes: None.

#### **4.2.9 Purchase Request Byte Stream (*PURCH\_REQ\_BYTE\_STREAM*)**

Description: The Purchase Request Byte Stream describes a purchase request transaction between the game station and the transaction processing system. It is used to request the purchase of one or more items, chances, plays, or tickets. The byte stream identifies the total cost of the purchase, the number of items purchased, and a [\*PURCH\\_ITEM\\_REQ\\_BYTE\\_STREAM\*](#) for each item in the purchase request.

Field Definitions:

[*total\_cost* U4] The total cost of the purchase as computed by the game station and based on the sum of the individual purchase requests.  
[*npurch* U4] This field specifies the number of items to be purchased. Also indicates the number of purchase item request byte streams that follow.

[\*PURCH\\_ITEM\\_REQ\\_BYTE\\_STREAM\*](#) <*npurch*>

Notes: For OGP version 2.2 and earlier, the purchase request byte stream is limited to purchases for a single game; that is, the game ID must be the same for each purchase in the purchase item request ([\*PURCH\\_ITEM\\_REQ\\_BYTE\\_STREAM\*](#)). The maximum number of items that can be purchased in one message is defined by the OGP\_MAX\_ORDER constant in the OGP header file (see [Appendix B](#)).

#### **4.2.10 Purchase Item Complete Byte Stream (*PURCH\_ITEM\_CMPLT\_BYTE\_STREAM*)**

Description: The Purchase Item Complete byte stream is used to describe the successful purchase of a single item, chance, play, or ticket. It echoes

the Purchase Item Request byte stream data and provides a unique item identifier and a payout field to provide payout information when the item purchased is a chance, play, or ticket.

Field Definitions:

PURCH\_ITEM\_REQ\_BYTE\_STREAM

[*item\_ID* U4] A unique item identifier associated with this item, chance, play, or ticket purchased.

[*payout* U4] The pay out (in pennies) if the item is a chance, play, or ticket. The *payout* field should be set to zero (0), if the item being purchased is not a chance, play, or ticket..

#### 4.2.11 Purchase Complete Byte Stream (PURCH\_CMPLT\_BYTE\_STREAM)

Description: The Purchase Complete Byte Stream is used to describe a completed purchase transaction between the game station and the transaction processing system. It provides the total cost of the purchase, the updated user account balance after the purchase, and the number of items, chances, plays, or tickets purchased. A PURCH\_ITEM\_CMPLT\_BYTE\_STREAM is included for each item, chance, play or ticket purchased.

Field Definitions:

ACCT\_UPDT\_BYTE\_STREAM

[*total\_cost* U4] The total cost of the purchase, as computed by the transaction processing system and based on the sum of the individual tickets purchased.

[*npurch* U4] The number of items, chances, plays, or tickets purchased.

PURCH\_ITEM\_CMPLT\_BYTE\_STREAM <*npurch*>

Notes: None.

#### 4.2.12 Game Play Byte Stream (GAME\_PLAY\_BYTE\_STREAM)

Description: The Game Play Byte Stream is used to indicate that a game play, game chance, or game ticket reveal has occurred. The byte stream

specifies the amount of the credit (win) or debit (loss) to be applied to the player's account balance, the amount wagered, the game ID, the item ID and any item specific data.

Field Definitions:

[ <i>game_id</i> C<16>]	A unique game identifier indicating the game associated with this transaction.
[ <i>item_id</i> U4]	The unique identifier associated with the chance, play, or ticket.
[ <i>amount</i> I4]	The <i>amount</i> field specifies the net transaction amount (credit or debit) in pennies. This field specifies the amount won or lost, after subtracting the wager amount. A debit is represented as a negative value.
[ <i>wager</i> U4]	The amount wagered in pennies. This field specifies the total amount of the bet, prior to any adjustment for a win or loss.
[ <i>item_data</i> B<64>]	Item specific data associated with the chance, play, or ticket. If the item data is a character string, the string must be terminated with a null (0) byte. If there is no item specific data, this field should be set to all null bytes.

Notes: For **purchase-with-play** type games, a loss is represented as zero (0). For **play-without-purchase** type games, a loss is represented as a negative number in the amount field and the *item\_id* and *item\_data* fields are set to zero (0). See section [6.3 Game Type Specific Protocols](#) for more information.

#### 4.2.13 Game Result Byte Stream (*GAME\_RESULT\_BYTE\_STREAM*)

Description: The Game Result Byte Stream is used to update the game station with the final results of a game play, chance, or game ticket reveal. It provides the updated account balance, echoes back the game play byte stream information from the preceding game play request, and provides result data.

Field Definitions:

GAME\_PLAY\_BYTE\_STREAM

[*result\_data* B<64>] Game specific result data associated with the winning or losing chance, play, or ticket. If there is no game specific result data, this field should be set to all null bytes.

[*promo\_id* U4] A non-zero value indicates that the Game Play was in conjunction with a special promotional offer and uniquely identifies the promotion.

Notes: None.

#### 4.2.14 Progressive Payout Byte Stream (*PROGRESSIVE\_BYTE\_STREAM*)

Description: The Progressive Payout Byte Stream is used to inform the Game Station of the current progressive jackpot for a specific *game\_id*.

Field Definitions:

[*game\_id* C<16>] The game identifier.

[*payout* U4] The progressive jackpot payout for this game in pennies.

[*msg* C<128>] Optional text message that may be displayed on the terminal marquee. The last byte of the message must be a null (0) byte (or the first byte must be a null byte if no message is being provided).

Notes: None.

#### 4.2.15 Game Statistics Byte Stream (*GAME\_STATS\_BYTE\_STREAM*)

Description: The Game Statistics Byte Stream provides game specific cumulative statistics on a per game station basis. The statistics are obtained for a specific game station from software or electronic meters physically located on the game station terminal.

Field Definitions:

[ <i>game_id</i> C<16>]	This field specifies the game identifier that indicates the type of game to which the statistics apply.
[ <i>wagers</i> U4]	The cumulative amount wagered (in pennies) on this game at the specified game station.
[ <i>won</i> U4]	The cumulative amount won (in pennies) on this game at the specified game station.
[ <i>jackpot</i> U4]	The cumulative amount won in progressive jackpots at this game station.
[ <i>nsold</i> U4]	The cumulative number of chances, plays, or tickets sold for this game at the specified game station.

Notes:               None.

#### 4.2.16 Game Station Statistics Byte Stream (*GS\_STATS\_BYTE\_STREAM*)

Description:       The Game Station Statistics Byte Stream provides game station summary statistics and game specific statistics data. The statistics are obtained from game station software or electronic meters physically located on the game station terminal.

Field Definitions:

[ <i>ndoor</i> U4]	The number of times the game station door has been opened.
[ <i>gs_jackpot</i> U4]	The cumulative amount won (in pennies) in progressive jackpots at this game station.
[ <i>gs_wagers</i> U4]	The cumulative amount wagered (in pennies) at this game station.
[ <i>gs_won</i> U4]	The cumulative amount won (in pennies) at this game station.
[ <i>gs_sold</i> U4]	The cumulative number of chances, plays, or tickets sold at this game station.
[ <i>ngames</i> U4]	The number of game statistic records contained in this byte stream.

[GAME\\_STAT\\_BYTE\\_STREAM](#)<ngames>

Notes: The maximum number of [GAME\\_STAT\\_BYTE\\_STREAM](#) records is defined by the OGP\_MAX\_GAME\_STATS constant in the OGP header file (see [Appendix B](#)).

#### 4.2.17 Status Byte Stream ([STATUS\\_BYTE\\_STREAM](#))

Description: The Status Byte Stream is used in transmitting Game Station status to the transaction processing system. The Status Byte Stream provides hardware and software level status information specific to system operation.

Field Definitions:

[*status* U4] This field specifies a status code that represents the current state of the game station. See section [5.3 Status Code Definitions](#) for specific status code values.

Notes: None.

#### 4.2.18 Acknowledge Message Byte Stream ([ACKMSG\\_BYTE\\_STREAM](#))

Description: The Acknowledge Message Byte Stream provides message acknowledgement status relative to game station and transaction processing system messages.

Field Definitions:

[*reason* U4] This field specifies the error code that indicates the reason for a negative acknowledgement. A non-zero value must be specified for this field for all negative Acknowledgement messages and Disable messages. For positive Acknowledgement messages, this field should be set to zero. See section [5.5 Error Code Definitions](#) for the list of error code values that may be specified.

[*minor\_code* U4] The minor error code used for negative acknowledgements. This field provides a secondary code that further defines the

*reason* for a negative acknowledgement. This field is optional, but must be set to 0 if not used.

[*msg* C<128>]

This field provides an optional text message that amplifies the *reason* field. The last byte of the message must be a null (0) byte (or the first byte must be a null byte if an explanatory message is not being provided).

Notes:

The OGP\_ACK\_POS flag is used to positively acknowledge receipt of a message from a peer and indicates that the message was successfully processed. The OGP\_ACK\_NEG flag is used to negatively acknowledge a message from a peer indicating that a specified action could not be performed. Additionally, a peer may specify that an Acknowledge message should not be sent in response to a message through the use of the OGP\_ACK\_NREQ flag. The *flags* field in the [MSG\\_HEADER\\_BYTE\\_STREAM](#) of a message received from a peer should be examined for the OGP\_ACK\_NREQ flag prior to sending an Acknowledge Message to the peer.

The Acknowledge message flags are set in the *flags* field of the Message Header Byte Stream. See section [5.4 Message Flag Definitions](#) for definitions of the Acknowledge Message related flags and their corresponding values.

#### 4.2.19 Acknowledged Message Type Byte Stream ([ACKMT\\_BYTE\\_STREAM](#))

Description:

The Acknowledged Message Type Byte Stream is used to convey information about the type of message being acknowledged. This byte stream is primarily intended for use by the transaction processing system, in order to map message protocols that are not acknowledge based to the OGP. Game Stations may default this byte stream to all zeroes (however, all of the bytes of this byte stream must be sent where specified).

Field Definitions:

[*mt* U4]

This field specifies the message identifier for the type of message being acknowledged.

Notes:

None.

#### 4.2.20 Win Level Configuration Byte Stream (*WLCONFIG\_BYTE\_STREAM*)

Description: The Win Level Configuration Byte Stream is used to transmit a single win-level in terms of its identifier, payout and the odds of winning at that payout level. The Win Level Configuration byte stream has the following format:

Field Definitions:

[ <i>win_level_id</i> U4]	The win-level identifier.
[ <i>flags</i> U4]	Win level flags.
[ <i>payout</i> U4]	Pay out in pennies.
[ <i>n</i> U4]	The total number of possible winning outcomes.
[ <i>m</i> U4]	Total possible outcomes at this level.

Notes: The *n* and *m* fields collectively represent the odds of winning at the specified payout amount; i.e. the player has *n* out of *m* odds of a win at this payout level.

#### 4.2.21 Game Configuration Byte Stream (*GMCONFIG\_BYTE\_STREAM*)

Description: The Game Configuration Byte Stream is used to download a game definition to the to a Game Station. A game configuration is defined in terms of a game identifier, type of game, the cost to play the game and the win-levels associated with the game. The Game Configuration byte stream has the following format:

Field Definitions:

[ <i>game_id</i> C<16>]	A unique game identifier indicating the game associated with this game definition.
[ <i>game_type</i> U4]	An unsigned integer value indicating the type of game in terms of purchase and play protocol. See <a href="#">Table 5.7 - Game Type Definitions</a> for the set of values that may be specified for this field and section <a href="#">6.3 Game Type Specific Protocols</a> for a discussion of game types.
[ <i>cost</i> U4]	The cost for this game in pennies.

[*nwinlevels* U4] Specifies the number of win-levels comprising this game definition.

WLCONFIG\_BYTE\_STREAM<*nwinlevels*>

Notes: None.

#### 4.2.22 Game Station Configuration Byte Stream (GSCONFIG\_BYTE\_STREAM)

Description: The Game Station Configuration Byte Stream is used to download the set of games that a Game Station will make available to a player. In addition, when using TCP/IP communications, this byte stream may optionally specify a new host IP address and/or port number to which the Game Station should reconnect.

If the flag OGP\_MSG\_INIT flag is set in the message header flags for the message carrying this byte stream, this configuration should overwrite the current Game Station configuration. If the OGP\_MSG\_INIT flag is not set, this configuration should be appended to the current Game Station configuration.

Note that due to maximum message size restrictions in the OGP (i.e. OGP\_MAX\_MSG\_SIZE), multiple OGP\_TPGS\_GSCONFIG messages may be sent by the transaction processing system in order to completely specify a game station configuration. The game station can use the *index* and *count* fields of this byte stream to aid in determining when the final OGP\_TPGS\_GSCONFIG message has been received.

##### Field Definitions:

[*ip\_addr* C<16>] If the first byte is not a null (0) byte, specifies the new IP address to which the Game Station should reconnect.

[*port* U4] If the *ip\_addr* field above is non-null, specifies the new port number to which the Game Station should reconnect.

[*index* U2] This field specifies the message index of the current OGP\_TPGS\_GSCONFIG message.

[*count* U2] Total number of OGP\_TPGS\_GSCONFIG messages being sent as part of the current Game Station configuration.

[*rsvd2* C<256>] Reserved for future use.  
[*ngames* U4] The number of game definitions specified in this configuration message.

[GMCONFIG\\_BYTE\\_STREAM](#)<*ngames*>

Notes: None.

#### 4.2.23 ATM Request Byte Stream (*ATM\_REQUEST\_BYTE\_STREAM*)

Description: The ATM Request Byte Stream is used to deposit funds into a specified player account, withdraw funds from a specified player account or to transfer funds between two different accounts owned by the same player.

Field Definitions:

[ACCTNO\\_BYTE\\_STREAM](#) Account number and PIN of the account to which the requested *action* is to be applied.

[ACCTNO\\_BYTE\\_STREAM](#) Account number and PIN of the account to/from which money is being transferred. This byte stream is ignored, if the requested *action* is not a transfer transaction type.

[*action* U4] This field specifies the type of ATM transaction to be performed. See section [5.8 ATM Transaction Code Definitions](#) for specific action code values.

[*amount* U4] Requested transaction amount (in pennies).

Notes: None.

#### 4.2.24 Player Rating Session Byte Stream (*PRATS\_SESSION\_BYTE\_STREAM*)

Description: The Player Rating Session Byte Stream is used to specify player rating and tracking information.

Field Definitions:

[ <i>source</i> C<8>]	This field is used to specify the manufacturer or system that generated the session data. This value is assigned as needed.
[ <i>acctno</i> C<32>]	The player account number associated with this session information. The last byte of the account number must be a null (0) byte.
[ <i>pos_id</i> C<16>]	A unique point-of-sale identifier indicating where this session took place.
[ <i>end_time</i> U4]	The session completion time expressed as seconds since 00:00:00 GMT January 1, 1970.
[ <i>length</i> U4]	The total session length in seconds.
[ <i>amt_spent</i> U4]	The total amount wagered in pennies.
[ <i>won</i> U4]	The total amount won in pennies.
[ <i>nitems</i> U4]	The number of items purchased or wagered.
[ <i>category</i> U2]	Specifies the type of the byte streams associated with the <i>Session_Data</i> information. See <a href="#">Table 5.9 Player Rating Category Code Definitions</a> for details.
[ <i>nsessions</i> U2]	Specifies the number of byte streams of <i>Session_Data</i> that follow.

***Session\_Data***<*nsessions*>

#### 4.2.25 Game Play Session Byte Stream (*PRATS\_GAME\_PLAY\_BYTE\_STREAM*)

Description: The Game Play Session Byte Stream is used to record game play activity for a specific game at a specific game station. It is a summary of player gaming activity for a single game within a single or multi-game player terminal.

##### Field Definitions:

[ <i>game_id</i> C<16>]	This field specifies a unique game identifier that indicates the type of game played by the customer during this session.
[ <i>nwagers</i> U4]	Total number of wagers for the specified game during this session.

- [*amt\_wagered* U4] Total amount wagered in pennies for the specified game during this session.
- [*amt\_won* U4] Total amount won in pennies for the specified game during this session.
- [*credit\_size* U2] Base credit size in pennies.
- [*nwager\_detail* U2] Number of byte streams of wager detail that follows.

**WAGER\_DETAIL\_BYTE\_STREAM**

**4.2.26 Wager Detail Byte Stream (WAGER\_DETAIL\_BYTE\_STREAM)**

Description: The Wager Detail Byte Stream provides wager detail information for a specific game.

Field Definitions:

- [*ncredits* U2] The number of credits wagered.
- [*nwagers* U2] The number of wagers at this credit level.

**4.2.27 Originating Message Byte Stream (ORIG\_MSG\_BYTE\_STREAM)**

Description: The Originating Message Byte Stream is used in negatively acknowledged messages to explicitly specify the message contents that could not be processed. The byte stream contains the original message content (i.e. the message bytes without the **START\_MSG\_BYTE\_STREAM** and **END\_MSG\_BYTE\_STREAM**) that was received and acknowledged by the peer.

The OGP\_INCL\_ACKD\_MSG message header flag should be set to indicate that this optional byte stream is included in the acknowledgment message. The Originating Message Byte Stream has the following format:

Field Definitions:

- [*msg\_size* U4] This field specifies the size of the message content data that follows.

[*orig\_msg* C<*msg\_size*>] The message content of the originating message.

#### **4.2.28 Draw Event Byte Stream (*DRAW\_EVENT\_BYTE\_STREAM*)**

Description: The Draw Event Byte Stream is used to describe a draw event.

Field Definitions:

[*event\_id* U4] Draw Event ID.

[*nvalues* U4] Specifies the number of draw event values that follow.

[*event\_values* U4<*nvalues*>] The array of event values (draw outcome).

Notes: Need to determine if “*event\_name* C<64>” should be added to the draw event byte stream.

#### **4.2.29 Draw Result Byte Stream (*DRAW\_RESULT\_BYTE\_STREAM*)**

Description: The Draw Result byte stream is used to pass the draw record information.

Field Definitions:

[*game\_id* U4] Game ID.

[*draw\_id* U4] Draw ID.

[*time* U4] Time that the draw record was generated. Expressed as seconds since 00:00:00 GMT January 1, 1970.

[*nevents* U4] The number of draw events that follow in the byte stream.

*DRAW\_EVENT\_BYTE\_STREAM* <*nevents*> Draw event information.

## 5 OGP FIELD CODE DEFINITIONS

### 5.1 Summary of Message Types

The following tables summarize the messages that are used to implement the Open Gaming Protocol. The ID column specifies the numeric message type identifiers (in decimal), which are provided in the *msg\_type* field of the Message Header Byte Stream. The S/U column indicates whether the message is solicited (S) or unsolicited (U) by the receiver. All values other than those listed in the tables below are considered invalid.

**Table 5.1.1 – Game Station to Transaction Processing System Messages**

Symbolic Message Name	ID	S/U	Description
OGP_GSTP_GSBOOT	1000	U	<a href="#">Game Station Boot Message</a> . This message is sent by the game station to request its initial game terminal configuration.
OGP_GSTP_STATUS	1001	SU	<a href="#">Game Station Status Message</a> . This message is used to send game station status information to the transaction processing system.
OGP_GSTP_LOGON	1002	U	<a href="#">Player Logon Request Message</a> . This message is used to initiate a player session at a gaming terminal.
OGP_GSTP_LOGOFF	1003	U	<a href="#">Player Logoff Request Message</a> . This message is used to terminate a player session at a gaming terminal.
OGP_GSTP_PURCHREQ	1004	U	<a href="#">Game Purchase Request Message</a> . This message is used to request the purchase of one or more chances, plays or tickets from the transaction processing system.
OGP_GSTP_GAME_PLAY	1005	U	<a href="#">Game Play Message</a> . This message is used to indicate that a game play or ticket reveal operation has occurred at the gaming terminal.

OGP_GSTP_USERPREF	1006	U	<a href="#">User Preferences Message</a> . This message is used to update the user's gaming preferences.
OGP_GSTP_GS_STATS	1007	S	<a href="#">Game Station Statistics Message</a> . This message is used to send game station statistics and meter readings to the transaction processing system.
OGP_GSTP_ACK	1008	S	<a href="#">Acknowledge Message</a> . This message is sent by the game station either to explicitly acknowledge receipt of a message from the transaction processing system or to indicate the failure in the processing of a request received from the transaction processing system.
OGP_GSTP_ATM_REQUEST	1009	U	<a href="#">ATM Transaction Request Message</a> . This message is sent by a game station or a kiosk to perform financial transactions on a player account.
OGP_GSTP_RATING	1010	U	<a href="#">Player Rating Message</a> . This message is used to send summary play data to the transaction processing system for player rating point calculation.

**Table 5.1.2 – Transaction Processing System to Game Station Messages**

<b>Symbolic Message Name</b>	<b>ID</b>	<b>S/U</b>	<b>Description</b>
OGP_TPGS_STATUSREQ	1	U	<a href="#">Status Request Message</a> . This message is used by the transaction processing system to request the current status of the game terminal.
OGP_TPGS_LOGOFF	2	U	<a href="#">Player Logoff Message</a> . This message is sent by the transaction processing system as a response to a game station Player Logon Request or to forcibly log a player off of a game station.
OGP_TPGS_PURCH	3	S	<a href="#">Purchase Message</a> . This message is sent by the transaction processing system to complete a Purchase Request transaction sent by the game station.

OGP_TPGS_LOGON	4	S	<a href="#">Player Logon Response Message</a> . This message is used by the transaction processing system to positively acknowledge a Player Logon Request Message and to send account information to a game station.
OGP_TPGS_GAME_RESULT	5	S	<a href="#">Game Result Message</a> . This message is sent by the transaction processing system to indicate the result of the preceding game play, chance or ticket reveal received from the game station.
OGP_TPGS_GS_STATSREQ	6	U	<a href="#">Game Station Statistics Request Message</a> . This message is sent by the transaction processing system to request current game statistics and kiosk meter readings.
OGP_TPGS_ACK	7	S	<a href="#">Acknowledge Message</a> . This message is sent by the transaction processing system either to explicitly acknowledge the receipt of a message from the game station or to indicate the failure in the processing of a request received from the game station.
OGP_TPGS_ENABLE	8	U	<a href="#">Game Station Enable Message</a> . This message is used by the transaction processing system to re-enable play at a game station.
OGP_TPGS_DISABLE	9	U	<a href="#">Game Station Disable Message</a> . This message is used by the transaction processing system to suspend play at a game station.
OGP_TPGS_GSCONFIG	10	SU	<a href="#">Game Station Configuration Message</a> . The transaction processing system sends this message in order to establish the game stations initial configuration or to reconfigure the game station on an as needed basis.
OGP_TPGS_TIME	11	U	<a href="#">Clock Synchronization Message</a> . This message is sent by the transaction processing system to force a game station to synchronize its system clock with the transaction processing system.

OGP_TPGS_PROGRESSIVE	12	U	<a href="#">Progressive Payout Message</a> . The transaction processing system sends this message to inform the game station that a progressive payout for a specific game has been updated.
OGP_TPGS_TERMNOTIFY	13	U	<a href="#">Terminal Notify Message</a> . The transaction processing system sends this message to instruct the terminal to display a message to the current player or on the terminal marquee.
OGP_TPGS_ACCT_UPDATE	14	S	<a href="#">Account Balance Update Message</a> . This message is sent by the transaction processing system in response to a game station ATM Request Message to complete the ATM transaction and to forward the updated player account information.

See [Appendix A – OGP Message Sequence Diagrams](#) for pictorial representations of the message sequences typically found in a system that implements the Open Gaming Protocol.

## 5.2 Account Number Type Definitions

Following are the set of values that may be provided for the *acctno\_type* field of an Account Number byte stream. All values other than those listed in the table below are considered invalid.

**Table 5.2 – Account Number Types**

Symbolic Name	Value	Description
OGP_ACCT_NO_NONE	0	The <i>acctno</i> field is empty. Used to indicate that a player is not currently logged onto the Game Station or that the transaction should apply to <b>any</b> player currently logged onto the Game Station.
OGP_ACCT_NO_CHAR128	2	The account number in the <i>acctno</i> field is valid and refers to a specific account. The <i>acctno</i> field is to be interpreted as an ASCII

		encoded character string.
--	--	---------------------------

### 5.3 Status Code Definitions

Following are the set of values that may be provided for the *status* field of a Game Station Status byte stream. All values other than those listed in the table below are considered invalid. Receipt of an unknown status value by the transaction processing system may cause the game station to be disabled.

**Table 5.3 – Game Station Status Codes**

Symbolic Name	Value	Description
OGP_ST_ON_LINE	0x00000000L	Situation is normal.
OGP_ST_TOO_INST	0x00000001L	Too many instant ticket types. Counters must be reset.
OGP_ST_MAINT	0x00000002L	Normal maintenance.
OGP_ST_TOO_PAY	0x00000003L	Paying too much.
OGP_ST_NO_COUNTER	0x00000004L	The counter memory is unavailable.
OGP_ST_DOOR_OPENED	0x00000005L	Main cabinet door opened.
OGP_ST_DOOR_CLOSED	0x00000006L	Main cabinet door closed.
OGP_ST_CAGE_OPENED	0x00000007L	Logic cage door opened.
OGP_ST_CAGE_CLOSED	0x00000008L	Logic cage door closed.
OGP_ST_BILL_VAL_OPENED	0x00000009L	Bill validator door opened
OGP_ST_BILL_VAL_CLOSED	0x00000010L	Bill validator door closed
OGP_ST_HOPPER_OPENED	0x00000011L	Coin hopper door opened
OGP_ST_HOPPER_CLOSED	0x00000012L	Coin hopper door closed
OGP_ST_SUPV_ACTIVATED	0x00000013L	Supervisor key switch activated, supervisor mode set.
OGP_ST_SUPV_CLOSED	0x00000014L	Supervisor mode exited.
OGP_ST_ATTENDANT_TOGGLED	0x00000015L	Attendant key switch activated.
OGP_ST_NETWORK_ERR	0x20000000L	Connection lost or closed due to a network error.
OGP_ST_SHUTDOWN	0x40000000L	Shutdown by system, the low 16-bits contain the reason as defined above.
OGP_ST_HARDWARE_ERR	0x80000000L	Hardware error. Lower 16 bits contain the H/W code (TBD).

## 5.4 Message Flag Definitions

Following are the values of the flags that may be provided for the *flags* field of the Message Header byte stream. Note that the OGP\_ACK\_NEG and OGP\_ACK\_POS flags are mutually exclusive and are applicable to Acknowledge messages only. Note that not all flags apply to all message types. All flag values other than those listed in the table below are silently ignored.

**Table 5.4 – Message Flags**

Symbolic Flag Name	Value	Description
OGP_ACK_NEG	0x00000001L	Message is negatively acknowledged.
OGP_ACK_POS	0x00000002L	Message is positively acknowledged.
OGP_ACK_NREQ	0x00000004L	Acknowledge not required. The sender requests that an Acknowledge message not be sent as a response.
OGP_INCL_ACKD_MSG	0x00000008L	Acknowledge message includes the message that is being acknowledged. This flag is intended only for internal use by the transaction processing system.
OGP_MSG_REPEAT	0x00000010L	Repeat Message. If this flag is set, this message is a reissue of a previous message.
OGP_MSG_UNSOLICITED	0x00000020L	Unsolicited message. This flag is a hint to the receiver, that the message is not a response to a message previously sent by the receiver. This flag is context sensitive and is not required for all unsolicited messages.
OGP_MSG_DUP_EVENT	0x00000040L	Indicates that the message is a duplicate event.
OGP_MSG_BANNER	0x00000100L	Banner Display. This flag is set if the message should be displayed in the terminal marquee, otherwise the message should be presented on-screen. The flag is only applicable to the <b>OGP_TPGS_TERMNOTIFY</b> message.

OGP_MSG_INIT	0x00001000L	Configuration Initialization. If this flag is set, replace the current configuration with the information contained in this message. Otherwise, append this information to the current configuration. This flag is context sensitive and is discussed where appropriate.
OGP_NON_GAME_TRANS	0x00010000L	If set, specifies message is a non-gaming service transaction.
OGP_MSG_QUICK_PICK	0x00100000L	If set, specifies that the play was the result of a quick-pick selection.
OGP_TXP_ORIGNTD_MSG	0x02000000L	If set, indicates that the transaction processor originated this message. This flag is intended only for internal use by the transaction processing system.
OGP_FEP_ORIGNTD_MSG	0x04000000L	If set, indicates that the front-end processor originated this message. This flag is intended only for internal use by the transaction processing system.

## 5.5 Error Code Definitions

Following are the values for the error codes that may be provided for the *reason* field of an Acknowledge Message byte stream. Note that these codes only have meaning for a negatively acknowledged message. Error codes other than those listed below have no meaning.

**Table 5.5 – OGP Error Codes**

<b>Symbolic Name</b>	<b>Value</b>	<b>Description</b>
OGP_NO_ERROR	0L	Operating condition is normal.
OGP_E_BAD_CHECKSUM	1L	Checksum of received message does not match checksum in message content.
OGP_E_INVALID_MSG_TYPE	2L	Unknown message type.
OGP_E_INVALID_MSG_SIZE	3L	Maximum OGP message size exceeded.

**PRINTED COPY IS UNCONTROLLED AND MAY BE OBSOLETE**

OGP2\_3-master.doc

OGP_E_FRAME_ERROR	4L	Ill-formed message sequence or frame word misalignment.
OGP_E_UNKNOWN_PROTOCOL	5L	Unknown message protocol or protocol version.
OGP_E_BAD_OGP_SEQUENCE	6L	Message serial number does not match expected value.
OGP_E_BAD_MSG_FORMAT	7L	Invalid message format; for example, string field not null-terminated/padded.
OGP_E_INVALID_MSG_FLAG	8L	Invalid usage of message flag.
OGP_E_CRYPT_FAILURE	9L	Encryption (or decryption) of data failed.
OGP_E_CRYPT_INIT_FAILURE	10L	Crypt initialization failure.
OGP_E_BAD_CRYPT_KEY	11L	Invalid encryption key.
OGP_E_BAD_CRYPT_IV	12L	Invalid encryption initial vector.
OGP_E_UNKNOWN_TERM_ID	15L	Terminal identifier is not recognized by the system.
OGP_E_UNKNOWN_TERM_ADDR	16L	Connection request attempted from a foreign address.
OGP_E_TERM_ID_IN_USE	17L	Terminal identifier already in use.
OGP_E_ALREADY_CONNECTED	18L	Connection from address already exists.
OGP_E_COMM_FAILURE	20L	Communications failure.
OGP_E_SVC_UNAVAILABLE	21L	Service currently unavailable.
OGP_E_HOST_UNREACHABLE	22L	Host address not reachable.
OGP_E_NET_UNREACHABLE	23L	Network not reachable.
OGP_E_DATA_NOT_AVAILABLE	24L	Data not available or connection closed.
OGP_E_UNEXPECTED_EOT	25L	Premature end of transmission or missing data.
OGP_E_NO_NEW_USERS	40L	Login capability is suspended.
OGP_E_NO_USER	41L	No user currently logged onto the Game Station.
OGP_E_NOT_LOGGED_ON	42L	Specified user is not logged onto Game Station.
OGP_E_MULTIPLE_LOGINS	43L	User is logged in at multiple stations.
OGP_E_NOT_SETTABLE	44L	User definition flag that was requested is not settable from a game station.
OGP_E_BAD_ACCTNO	60L	Unknown account number or invalid account number format.
OGP_E_BAD_ACCTNO_TYPE	61L	Unknown account number type.
OGP_E_ACCT_DISABLED	62L	Account has been disabled.

**PRINTED COPY IS UNCONTROLLED AND MAY BE OBSOLETE**

OGP2\_3-master.doc

OGP_E_ACCT_RESTRICTED	63L	Account has been restricted and is not allowed on this station.
OGP_E_ACCT_AT_LIMIT	64L	Account has reached credit limit.
OGP_E_PIN_NOT_SET	65L	A PIN has not been assigned to the account.
OGP_E_PIN_MISMATCH	66L	Incorrect PIN entry.
OGP_E_BALANCE_MISMATCH	67L	Game station and transaction processor account balances differ.
OGP_E_ACCT_ALREADY_EXISTS	68L	Account already exists.
OGP_E_ACCT_PLAY_EXPIRED	69L	The account play date has expired.
OGP_E_ACCT_REDEEM_EXPIRED	70L	The account redemption date has expired.
OGP_E_BAD_CARD_TYPE	71L	Invalid card number or account type/card type mismatch.
OGP_E_WAGER_EXCEEDS_BAL	72L	Wager exceeds current account balance.
OGP_E_INVALID_ACCT_STATUS	73L	Invalid account status.
OGP_E_BAL_EXCEEDS_LIMITS	74L	Balance update exceeds account limits.
OGP_E_BAD_GAME_ID	80L	The game ID is not active.
OGP_E_ORDER_TOO_LARGE	81L	Maximum order size exceeded.
OGP_E_ORDER_TOO_COSTLY	82L	Order exceeds maximum cost.
OGP_E_UNKNOWN_ORDER_ID	83L	The order ID is not recognized by the transaction processor.
OGP_E_BAD_GAME_DEF	84L	Invalid game definition.
OGP_E_PAY_TABLE_MISMATCH	85L	Pay table does not match internal software.
OGP_E_NO_TICKETS	86L	No tickets or chances available.
OGP_E_GS_ERROR	100L	Game station error – must set <i>minor_code</i> field to real game station error.
OGP_E_BAD_STATUS	101L	Current game station status can't support the requested command.
OGP_E_DISABLED	102L	The request won't be handled because the game station is disabled.
OGP_E_CLOCK_ERROR	103L	The game station's clock could not be satisfactorily set.
OGP_E_INTERNAL_ERROR	200L	Internal error – should never happen if code is correct.
OGP_E_INTERNAL_OVERFLOW	201L	Maximum internal value or table capacity exceeded.
OGP_E_NO_MEMORY	202L	Out of memory.

OGP_E_NO_MORE_RESOURCES	203L	System resources exhausted (i.e. sockets, shared memory, semaphores, etc.).
OGP_E_NO_PERMISSION	204L	No permission to perform requested operation.
OGP_E_INVALID_PARAM	205L	Invalid message parameter.
OGP_E_INVALID_TX	206L	Invalid transaction.
OGP_E_NOT_IMPLEMENTED	207L	Functionality not implemented.
OGP_E_TX_FAILED	208L	Transaction processing failed.
OGP_E_TX_ARCHIVE_FAILED	209L	Required transaction archiving failed.

## 5.6 User Information Flag Definitions

Following are the set of values that may be provided for the *pflags* field of a User Preferences byte stream. All flag values other than those listed in the table below are silently ignored. Multiple flag values may be arithmetically ORed together.

**Table 5.6 – User Information Flags**

<b>Symbolic Flag Name</b>	<b>Value</b>	<b>Meaning</b>
OGP_UF_SHOWNAME	0x00000001L	User likes name displayed.
OGP_UF_PREFERRED	0x00000002L	User is a preferred customer.
OGP_UF_SILENT	0x00000004L	User prefers no sound.
OGP_UF_LEFT_HAND	0x00000008L	User prefers left hand.

## 5.7 Game Type Definitions

The following table identifies the set of values that can be provided in the *game\_type* field of the Game Configuration byte stream. All values other than those listed in the table below are considered invalid.

**Table 5.7 – Game Types**

<b>Symbolic Flag Name</b>	<b>Value</b>	<b>Meaning</b>
OGP_GT_PURCHWPLAY	0x00000001L	Purchase with Play type of game.

OGP_GT_PURCHWOPLAY	0x00000002L	Purchase without Play type of game.
OGP_GT_PLAYWOPURCH	0x00000003L	Play without Purchase type of game.

## 5.8 ATM Transaction Code Definitions

The following table identifies the set of values that can be provided in the *action* field of the ATM Request byte stream. All values other than those listed in the table below are considered invalid.

**Table 5.8 – ATM Transaction Type Codes**

<b>Symbolic Flag Name</b>	<b>Value</b>	<b>Meaning</b>
OGP_ATM_BALANCE_INQUIRY	0x00000000L	Perform a balance inquiry.
OGP_ATM_DEPOSIT	0x00000001L	Deposit money into an account.
OGP_ATM_WITHDRAWAL	0x00000002L	Withdraw money from an account.
OGP_ATM_XFER_FROM_OTHER	0x00000003L	Transfer money from another account owned by the same player.
OGP_ATM_XFER_TO_OTHER	0x00000004L	Transfer money to another account owned by the same player.

## 5.9 Player Rating Category Code Definitions

The following table identifies the set of values that can be provided in the *category* field of the Player Rating Session Byte Stream. All values other than those listed in the table below are considered invalid.

**Table 5.9 – Player Rating Category Type Codes**

<b>Symbolic Flag Name</b>	<b>Value</b>	<b>Session Data Byte Stream Type</b>
OGP_PRATS_CATEGORY_SLOT	0x00000001L	<b><i>PRATS_GAME_PLAY_BYTE_STREAM</i></b>

## **6 OGP COMMUNICATION REQUIREMENTS**

The OGP is a byte stream protocol for use in a cashless transaction gaming system. The protocol defines the set of messages that may be passed between a *game station* and a *transaction processor*. The protocol may be implemented over RS-232 or TCP/IP based connections. The following sections describe specific communication requirements.

### **6.1 RS-232 Serial Communications Protocol**

Game terminals implementing the Open Gaming Protocol using RS-232 serial protocol shall establish a connection using the following communications settings: 9600 baud, no parity, eight (8) data bits, and one (1) stop bit.

### **6.2 TCP/IP Communications Protocol**

**Old:** When using TCP/IP communications, ports 12100-12199 are reserved for use by the OGP protocol, with port 12100 being the default or *well-known* port number on which a connection to the Transaction Processor should first be established. The transaction processing system may reassign a host address and/or port number for a Game Station to use as part of a Game Station Configuration message (see section 3). The Game Station should re-establish a connection using the host address and port number assigned by the transaction processing system.

**Alternate:** When using TCP/IP communications, the Game Station connects to the transaction processing system through a port number assigned by the system administrator. Currently, ports 31000-31999 are reserved for use by the transaction processing system. Game Stations establish connections on even numbered ports. Odd numbered ports are reserved for private use by the transaction processing system. Additionally, the transaction processing system may reassign a host address and/or port number for a Game Station to use as part of a Game Station Configuration message (see

section 3). If provided, the Game Station should re-establish a connection using the host address and port number assigned by the transaction processing system.

## **6.3 Game Type Specific Protocols**

Game type specific protocol refers to the game specific purchase and play interactions between the game station and transaction processing system. Version 2.3 of the protocol supports three Game Type specific protocols: 1) Purchase with Play, 2) Purchase without Play, and 3) Play without Purchase.

### **6.3.1 Purchase with Play**

This protocol is a two-step sequence that requires the game station 1) issue and complete a Purchase Request transaction and 2) issue and complete a Game Play transaction to complete a game play sequence. See **Appendix A Message Sequence Diagram 1** for a pictorial representation of this type of transaction. This game type supports a game chance, play, or ticket whose outcome is determined by the transaction processing system.

### **6.3.2 Purchase without Play**

This protocol requires the game station issue and complete a Purchase Request transaction to complete a game play sequence. The outcome of the purchased game chance, play, or ticket is predetermined by the transaction processing system. When purchased, the outcome is provided back to the game station as part of the purchase transaction. The game station is responsible for revealing the outcome and updating account balance information displayed to the player. See **Appendix A Message Sequence Diagram 2** for a pictorial representation of this type of transaction. This game type supports finite deck games, such as Scratch Ticket games, where all the winning and losing tickets are predetermined and randomly dispensed to the game stations.

### **6.3.3 Play without Purchase**

This protocol requires no purchase transaction. The game station determines the outcome of the game play and issues a Game Play message to the transaction processing system notifying the transaction processing system of the game play and outcome. The transaction processing system responds with updated account balance information. See **Appendix A Message Sequence Diagram 3** for a pictorial representation of this type of transaction. This game type supports game server or game station determined outcome games.

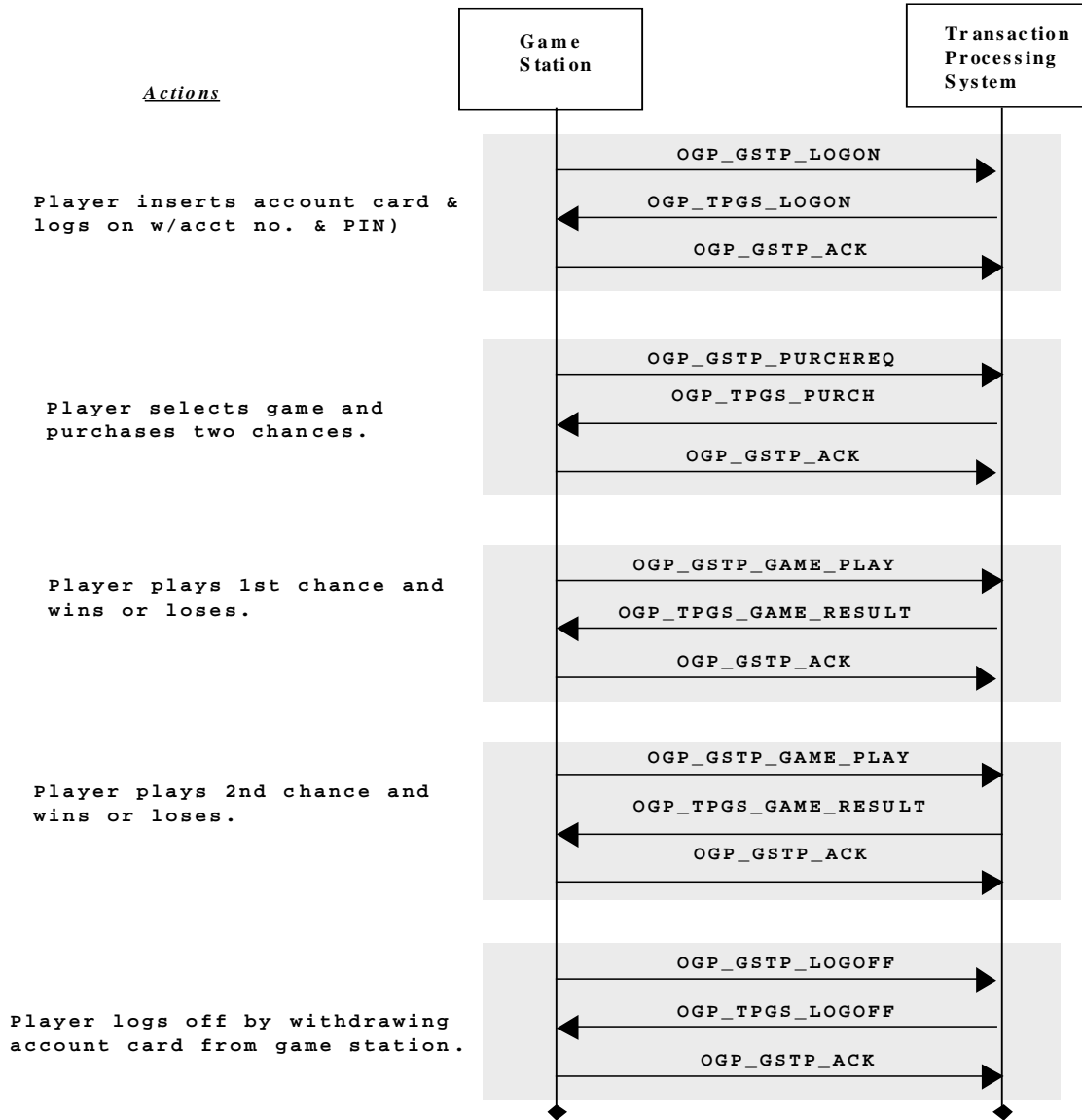
## **6.4 Acknowledge Protocol**

Unless specifically noted, all messages in the OGP are acknowledged either explicitly through the use of an Acknowledge message, or implicitly through the receipt of the appropriate response message type, regardless of the sender. An Acknowledge message, however, terminates a peer-to-peer message sequence and should never be acknowledged.

All messages shall be acknowledged within a five (5) second time period. If an acknowledgement or a response is not received within the specified time period, the sender can assume a message transmission error and can (optionally) retransmit the same message. Messages should be retransmitted with the `OGP_MSG_REPEAT` flag set. At no time shall a sender originate a new message prior to the receipt of an acknowledgement from a previous message. However, a sender may respond to an unsolicited message from its peer while waiting for a response from its most recently transmitted message.

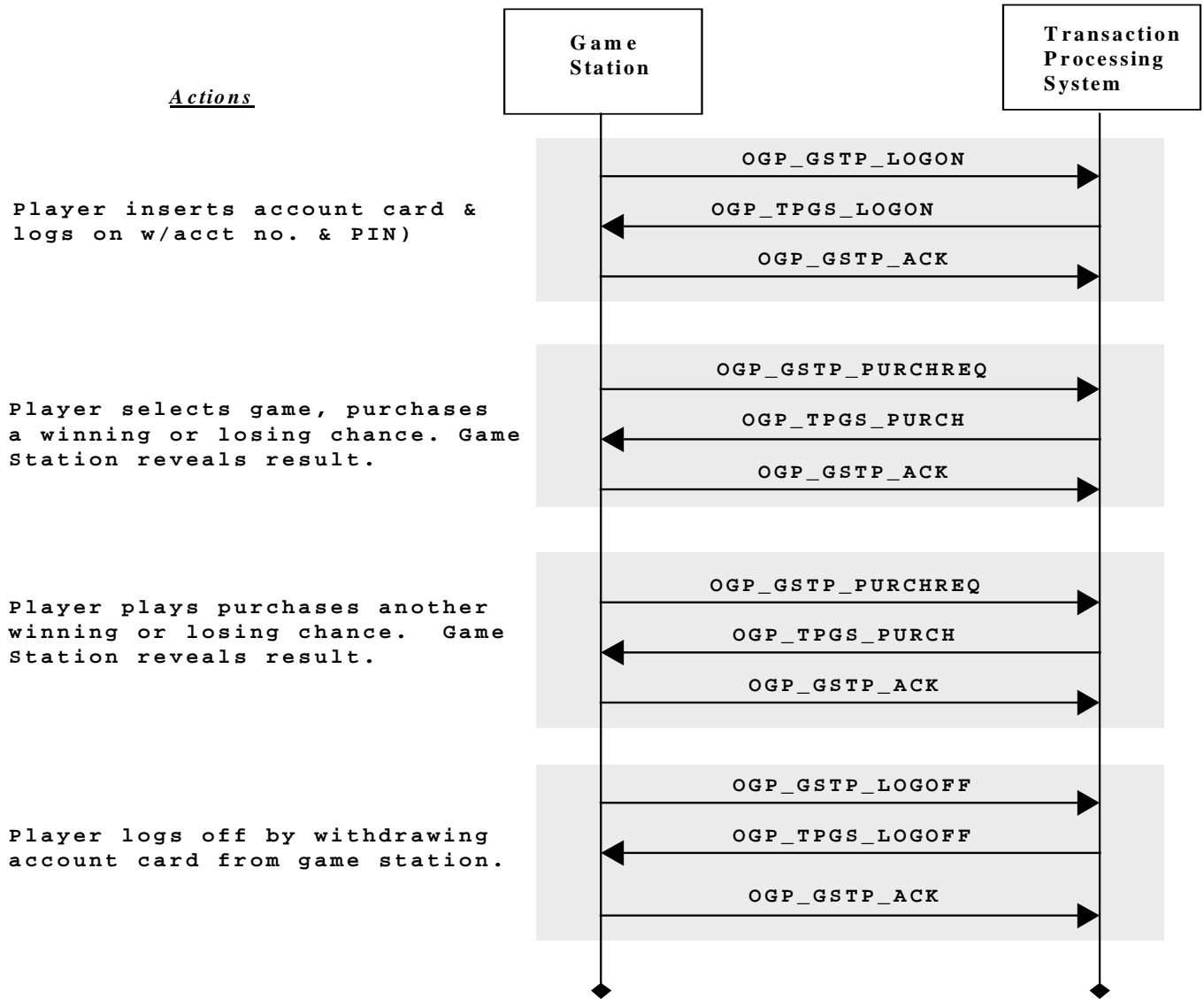
## **APPENDIX A - OGP MESSAGE SEQUENCE DIAGRAMS**

Message Sequence Diagram 1  
 Player Logon, Purchase, Play, and Logoff transactions\*



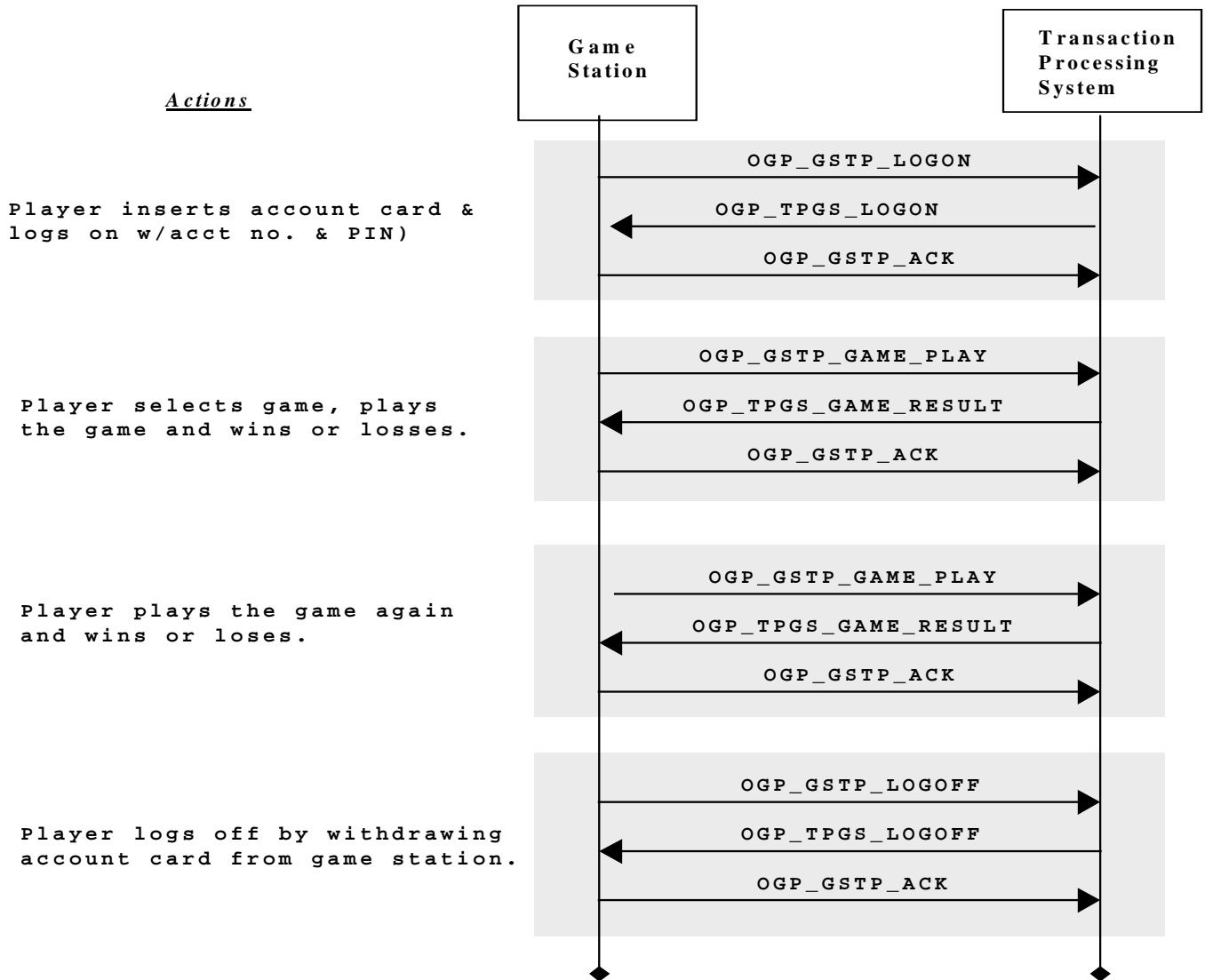
\*Assumes purchase with play type games.

**Message Sequence Diagram 2  
Player Logon, Purchase, and Logoff transactions\***



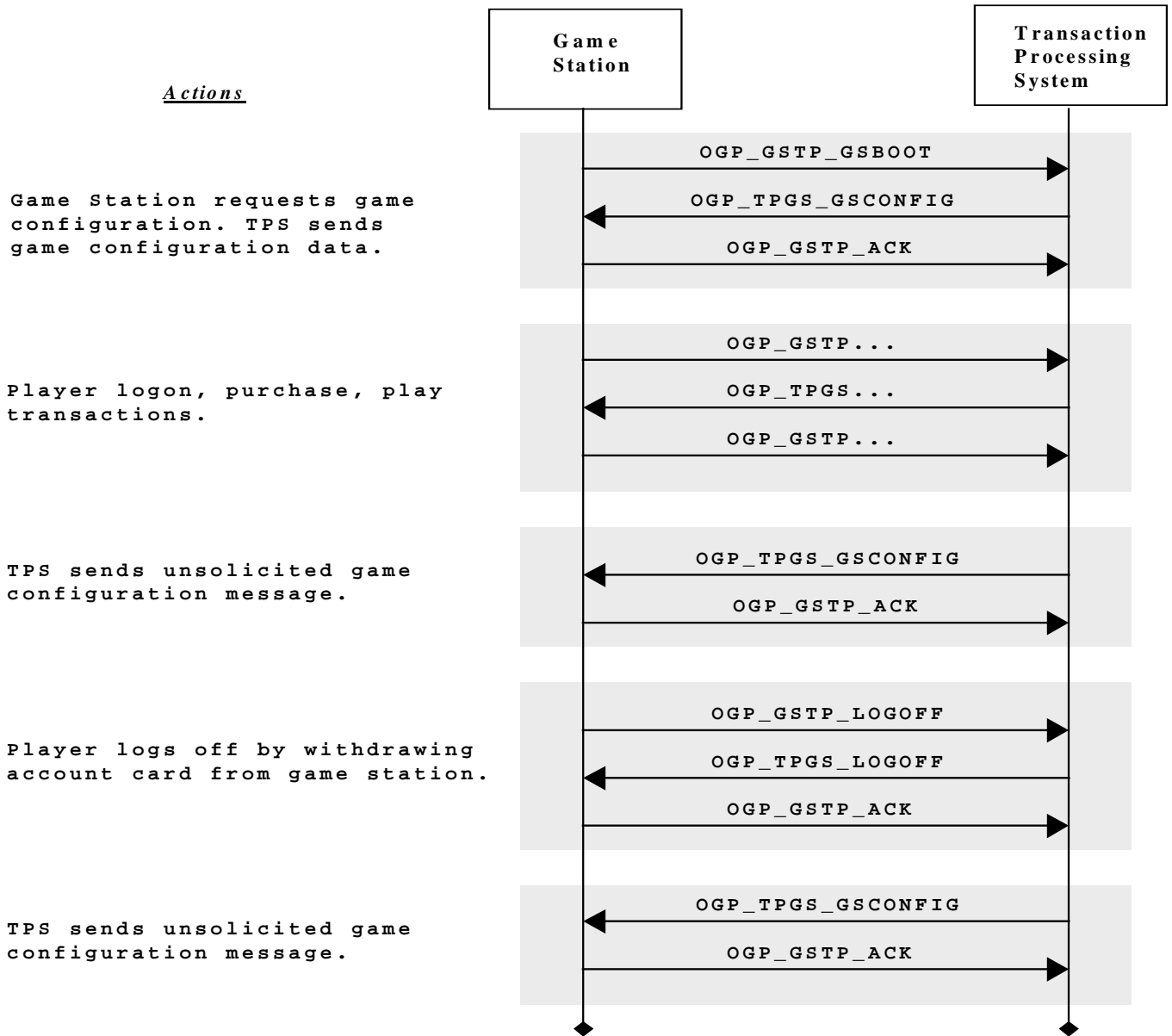
\*Assumes purchase w/o play type games.

**Message Sequence Diagram 3  
Player Logon, Play, and Logoff transactions\***

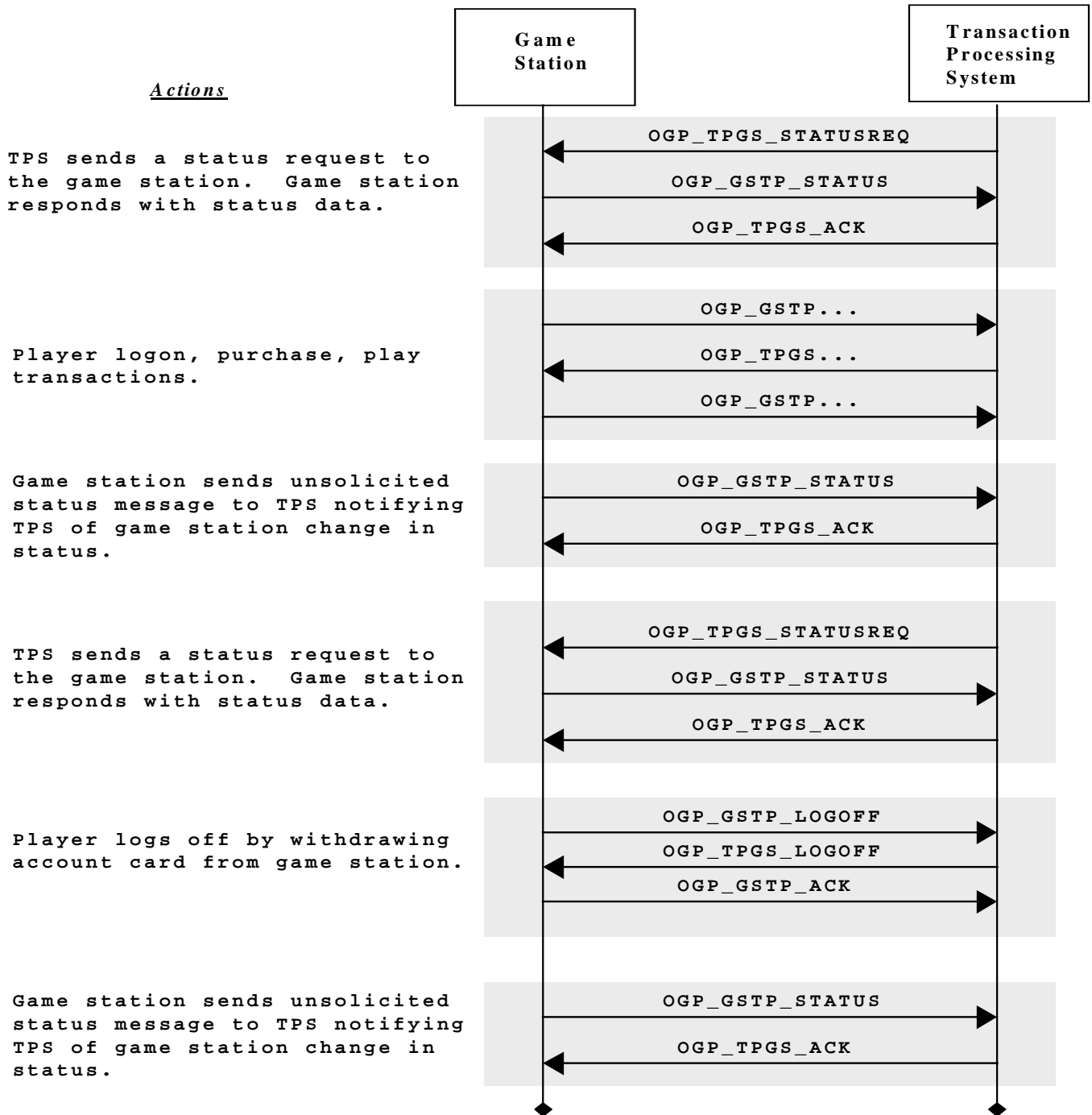


\*Assumes play w/o purchase type games.

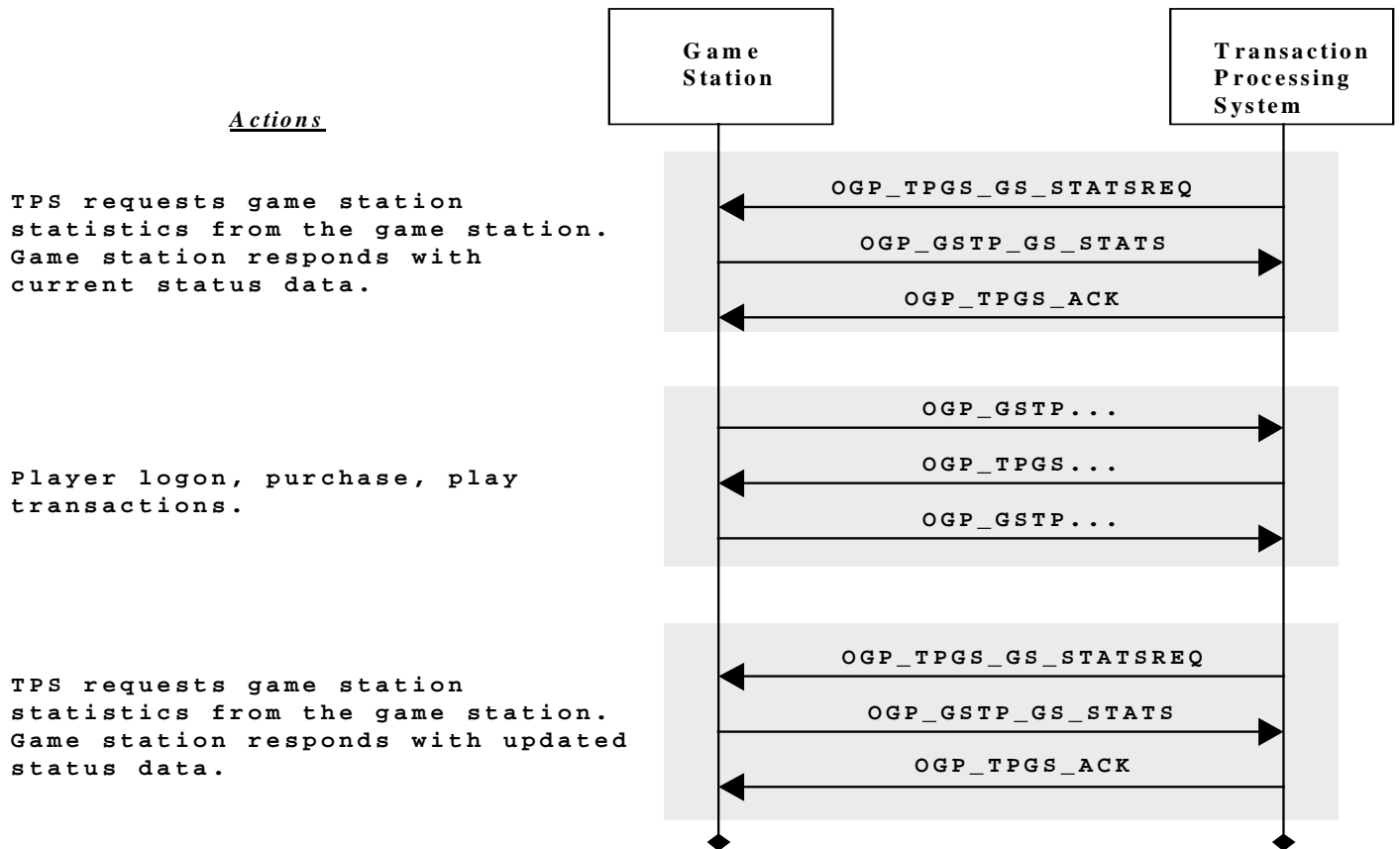
**Message Sequence Diagram 4**  
**Game Station Boot and Configuration Transactions**



**Message Sequence Diagram 5  
Status Message Transactions**



Message Sequence Diagram 6  
Game Statistics Message Transactions



## APPENDIX B - OGP C/C++ LANGUAGE COMPATIBLE DEFINITIONS

```
/*
** FILE:          OGP.h
**
** TYPE:          C/C++ HEADER FILE
**
** DESCRIPTION:   Definitions for the Open Gaming Protocol.
**
*/

#ifndef OGP_H_INCLUDED
# define OGP_H_INCLUDED

/*
 * Version Definitions
 */

#define OGP_PROTOCOL_MAJOR_VERSION      2
#define OGP_PROTOCOL_MINOR_VERSION     3
#define OGP_PROTOCOL_RELEASE           0
#define OGP_PROTOCOL_VERSION_STRING    "Open Gaming Protocol v2.3.0"

/*
 * Communications protocols
 */

#define PROTOCOL_OGP                    0x4f47

/*
 * Message type identifiers
 */

/* Transaction Processor to Game Station */
#define OGP_TPGS_STATUSREQ              1L /* Status request */
#define OGP_TPGS_LOGOFF                 2L /* User log-off */
#define OGP_TPGS_PURCH                   3L /* Filled purchase request */
#define OGP_TPGS_LOGON                  4L /* User logon response */
#define OGP_TPGS_GAME_RESULT            5L /* Game result */
#define OGP_TPGS_GS_STATSREQ            6L /* GS statistics request */
#define OGP_TPGS_ACK                    7L /* Acknowledge */
#define OGP_TPGS_ENABLE                  8L /* GS Enable */
#define OGP_TPGS_DISABLE                 9L /* GS Disable */
#define OGP_TPGS_GSCONFIG               10L /* GS Configuration */
#define OGP_TPGS_TIME                    11L /* Clock Sync */
#define OGP_TPGS_PROGRESSIVE            12L /* Progressive pay out */
#define OGP_TPGS_TERMNOTIFY             13L /* Terminal notification */
#define OGP_TPGS_ACCT_UPDATE            14L /* Account balance update */
#define OGP_FIRST_TP_MSG_TYPE           OGP_TPGS_STATUSREQ
#define OGP_LAST_TP_MSG_TYPE            OGP_TPGS_ACCT_UPDATE

/* Draw Service (via Transaction Processor) to Game Station */
#define OGP_TPGS_DRAWSERVICE            99L /* Draw Service Message */

/* Game Station to Transaction Processor */
#define OGP_GSTP_GSBOOT                 1000L /* Game station boot */
#define OGP_GSTP_STATUS                  1001L /* Status */
#define OGP_GSTP_LOGON                   1002L /* User log-on */
#define OGP_GSTP_LOGOFF                  1003L /* User log-off */
```

**PRINTED COPY IS UNCONTROLLED AND MAY BE OBSOLETE**

OGP2\_3-master.doc

```
#define OGP_GSTP_PURCHREQ      1004L /* Purchase request */
#define OGP_GSTP_GAME_PLAY    1005L /* Game play */
#define OGP_GSTP_USERPREF     1006L /* Update user preferences */
#define OGP_GSTP_GS_STATS     1007L /* Game station statistics */
#define OGP_GSTP_ACK          1008L /* Acknowledge */
#define OGP_GSTP_ATM_REQUEST  1009L /* ATM transaction request */
#define OGP_GSTP_RATING       1010L /*Player Rating Session */
#define OGP_FIRST_GS_MSG_TYPE OGP_GSTP_GSBOOT
#define OGP_LAST_GS_MSG_TYPE  OGP_GSTP_RATING

/*
 * Message Flags
 */

/* Acknowledge message flags */
#define OGP_ACK_NEG          0x00000001L /* Acknowledge is negative */
#define OGP_ACK_POS          0x00000002L /* Acknowledge is positive */
#define OGP_ACK_NREQ         0x00000004L /* Acknowledge not required */
#define OGP_INCL_ACKD_MSG    0x00000008L /* Message being ackd is included */

/* General message flags */
#define OGP_MSG_REPEAT       0x00000010L /* Flag is set if message is a
/* reissue of a previous msg */
#define OGP_MSG_UNSOLICITED 0x00000020L /* Flag indicating that the
/* message was not solicited by
/* the receiver.
#define OGP_MSG_DUP_EVENT    0x00000040L /* Message is a duplicate event.
#define OGP_MSG_BANNER       0x00000100L /* Display message in marquee
#define OGP_MSG_INIT         0x00001000L /* Initial configuration or
/* reconfiguration, otherwise
/* append to previous config.
#define OGP_NON_GAME_TRANS   0x00010000L /* If set, specifies message is
/* a non-gaming service
/* transaction.
#define OGP_MSG_QUICK_PICK   0x00100000L /* If set, specifies that the
/* play was the result of a
/* quick-pick selection.
#define OGP_TXP_ORIGNTD_MSG  0x02000000L /* If set, TXP sent this message
#define OGP_FEP_ORIGNTD_MSG  0x04000000L /* If set, FEP sent this message

/*
 * Win level configuration flags
 */

#define OGP_WLVL_COMBINED    0x00000001L /* Combination win-level
#define OGP_WLVL_HAND_PAY    0x00000002L /* Win-level requires hand pay

/*
 * Error Codes
 */

#define OGP_NO_ERROR          0L /* Operating condition is normal */

/* Message header or message format error codes */
#define OGP_E_BAD_CHECKSUM    1L /* Incorrect checksum
#define OGP_E_INVALID_MSG_TYPE 2L /* Invalid message type
#define OGP_E_INVALID_MSG_SIZE 3L /* Invalid message size
#define OGP_E_FRAME_ERROR     4L /* Ill-formed message sequence
#define OGP_E_UNKNOWN_PROTOCOL 5L /* Unknown message protocol
#define OGP_E_BAD_OGP_SEQUENCE 6L /* Message serial number does
/* not match expected value
#define OGP_E_BAD_MSG_FORMAT  7L /* Invalid message format
#define OGP_E_INVALID_MSG_FLAG 8L /* Invalid message flag
#define OGP_E_CRYPT_FAILURE    9L /* Encryption (or decryption)
```

**PRINTED COPY IS UNCONTROLLED AND MAY BE OBSOLETE**

OGP2\_3-master.doc

```

/* of data failed */
#define OGP_E_CRYPT_INIT_FAILURE 10L /* Crypt initialization failure */
#define OGP_E_BAD_CRYPT_KEY 11L /* Bad encryption key */
#define OGP_E_BAD_CRYPT_IV 12L /* Bad encryption initial vector */

/* Connection error codes */
#define OGP_E_UNKNOWN_TERM_ID 15L /* Unknown terminal identifier */
#define OGP_E_UNKNOWN_TERM_ADDR 16L /* Unknown terminal address */
#define OGP_E_TERM_ID_IN_USE 17L /* Terminal ID already in use */
#define OGP_E_ALREADY_CONNECTED 18L /* Connection from address
/* already exists. */

/* Communications failure error codes */
#define OGP_E_COMM_FAILURE 20L /* Communications failure */
#define OGP_E_SVC_UNAVAILABLE 21L /* Service currently unavailable */
#define OGP_E_HOST_UNREACHABLE 22L /* Host address not reachable */
#define OGP_E_NET_UNREACHABLE 23L /* Network not reachable */
#define OGP_E_DATA_NOT_AVAILABLE 24L /* Data not available or EOF */
#define OGP_E_UNEXPECTED_EOT 25L /* Premature end of transmission */

/* User/login error codes */
#define OGP_E_NO_NEW_USERS 40L /* Login capability is suspended */
#define OGP_E_NO_USER 41L /* No user on the GS */
#define OGP_E_NOT_LOGGED_ON 42L /* Specified user is not on GS */
#define OGP_E_MULTIPLE_LOGINS 43L /* User is logged in at multiple
/* stations */
#define OGP_E_NOT_SETTABLE 44L /* User definition flag that
/* was requested is not
/* settable from a GS */

/* Account related errors */
#define OGP_E_BAD_ACCTNO 60L /* Invalid account number */
#define OGP_E_BAD_ACCTNO_TYPE 61L /* Invalid account number type */
#define OGP_E_ACCT_DISABLED 62L /* Account has been disabled */
#define OGP_E_ACCT_RESTRICTED 63L /* Account has been restricted
/* and is not allowed on this
/* station */
#define OGP_E_ACCT_AT_LIMIT 64L /* Account credit limit reached */
#define OGP_E_PIN_NOT_SET 65L /* PIN not assigned to account */
#define OGP_E_PIN_MISMATCH 66L /* Incorrect PIN entry */
#define OGP_E_BALANCE_MISMATCH 67L /* GS and transaction processor
/* account balances differ */
#define OGP_E_ACCT_ALREADY_EXISTS 68L /* Account Already Exists */
#define OGP_E_ACCT_PLAY_EXPIRED 69L /* Account Play Date expired */
#define OGP_E_ACCT_REDEEM_EXPIRED 70L /* Account Redempt. Date expired */
#define OGP_E_BAD_CARD_TYPE 71L /* Invalid card number or account*/
/* type/card type mismatch */
#define OGP_E_WAGER_EXCEEDS_BAL 72L /* Wager exceeds account balance */
#define OGP_E_INVALID_ACCT_STATUS 73L /* Invalid account status */
#define OGP_E_BAL_EXCEEDS_LIMITS 74L /* Balance update exceeds account*/
/* limit. */

/* Game transaction error codes */
#define OGP_E_BAD_GAME_ID 80L /* The Game ID is not in play */
#define OGP_E_ORDER_TOO_LARGE 81L /* Ticket order > MAX_ORDER */
#define OGP_E_ORDER_TOO_COSTLY 82L /* Ticket order cost > max cost */
#define OGP_E_UNKNOWN_ORDER_ID 83L /* The order ID was not
/* recognized by the TP */
#define OGP_E_BAD_GAME_DEF 84L /* Invalid game definition */
#define OGP_E_PAY_TABLE_MISMATCH 85L /* Pay table does not match
/* internal software */
#define OGP_E_NO_TICKETS 86L /* No tickets or chances avail. */

/* Game station error codes */
#define OGP_E_GS_ERROR 100L /* Game station error; must set
/* minor_code field to real
/* GS error code */
```

**PRINTED COPY IS UNCONTROLLED AND MAY BE OBSOLETE**

OGP2\_3-master.doc

```
#define OGP_E_BAD_STATUS          101L /* Current status can't support */
/* the requested command */
#define OGP_E_DISABLED           102L /* The request won't be handled */
/* because the GS is disabled */
#define OGP_E_CLOCK_ERROR        103L /* The GS's clock could not be */
/* satisfactorily set */

/* Generic error codes - requires amplification using minor_code and/or msg */
#define OGP_E_INTERNAL_ERROR     200L /* Internal error; should never */
/* happen if code is correct */
#define OGP_E_INTERNAL_OVERFLOW  201L /* Maximum internal value or */
/* table capacity exceeded */
#define OGP_E_NO_MEMORY          202L /* Out of memory */
#define OGP_E_NO_MORE_RESOURCES  203L /* System resources exhausted */
/* (i.e. sock, shm, sem, etc.) */
#define OGP_E_NO_PERMISSION      204L /* No permission to perform op */
#define OGP_E_INVALID_PARAM      205L /* Invalid message parameter */
#define OGP_E_INVALID_TX         206L /* Invalid transaction */
#define OGP_E_NOT_IMPLEMENTED    207L /* Function not implemented */
#define OGP_E_TX_FAILED          208L /* Transaction processing failed */
#define OGP_E_TX_ARCHIVE_FAILED  209L /* Transaction archiving failed */

/* Special return codes used by the Transaction Processing System only */
#define OGP_FAIL_WITHOUT_NACK    1000L
#define OGP_CLOSE_CONNECTION    1001L
#define OGP_SEND_NOOP           1002L
#define OGP_SEND_ONLINE_STATUS  1003L
#define OGP_SEND_PSEUDO_ACK      1004L
#define OGP_SEND_PSEUDO_GSCONFIG 1005L
#define OGP_INCOMPLETE_TRANSFER 1006L

/*
 * User Flags
 */
#define OGP_UF_SHOWNAME         0x00000001L /* User likes name displayed */
#define OGP_UF_PREFERED         0x00000002L /* Preferred customer */
#define OGP_UF_SILENT           0x00000004L /* Prefers no sound */
#define OGP_UF_LEFT_HAND        0x00000008L /* Prefers left hand */

/*
 * Game Types
 */
#define OGP_GT_PURCHWPLAY       1L /* Purchase with play type */
#define OGP_GT_PURCHWOPLAY      2L /* Purchase without play type */
#define OGP_GT_PLAYWOPURCH      3L /* Play without purchase type */

/*
 * Account Number Types
 */
#define OGP_ACCT_NO_NONE        0L
#define OGP_ACCT_NO_CHAR128     2L

/*
 * Status Codes
 */
#define OGP_ST_ON_LINE          0x00000000L /* Situation is normal. */
#define OGP_ST_TOO_INST         0x00000001L /* Shutdown, too many */
/* instant ticket types - */
```

**PRINTED COPY IS UNCONTROLLED AND MAY BE OBSOLETE**

OGP2\_3-master.doc

```
/* counters must be reset. */
#define OGP_ST_MAINT 0x00000002L /* Normal maintenance. */
#define OGP_ST_TOO_PAY 0x00000003L /* Paying too much. */
#define OGP_ST_NO_COUNTER 0x00000004L /* The Counter Memory is
/* unavailable. */
#define OGP_ST_DOOR_OPENED 0x00000005L /* Main cabinet door open */
#define OGP_ST_DOOR_CLOSED 0x00000006L /* Main cabinet door closed */
#define OGP_ST_CAGE_OPENED 0x00000007L /* Logic cage door open */
#define OGP_ST_CAGE_CLOSED 0x00000008L /* Logic cage door closed */
#define OGP_ST_BILL_VAL_OPENED 0x00000009L /* Bill validator opened */
#define OGP_ST_BILL_VAL_CLOSED 0x00000010L /* Bill validator closed */
#define OGP_ST_HOPPER_OPENED 0x00000011L /* Coin hopper opened */
#define OGP_ST_HOPPER_CLOSED 0x00000012L /* Coin hopper closed */
#define OGP_ST_SUPV_ACTIVATED 0x00000013L /* Supervisor key-switch
/* activated, supervisor
/* mode set
/* Supervisor mode exited
#define OGP_ST_SUPV_CLOSED 0x00000014L /* Supervisor mode exited */
#define OGP_ST_ATTENDANT_TOGGLED 0x00000015L /* Attendant key-switch
/* activated
/* Hardware error, low 16
/* bits are H/W code (TBD)
#define OGP_ST_HARDWARE_ERR 0x80000000L /* Hardware error, low 16
/* bits are H/W code (TBD)
#define OGP_ST_SHUTDOWN 0x40000000L /* Shutdown by system, low
/* 16 bits contain the
/* reason as defined above.
#define OGP_ST_NETWORK_ERR 0x20000000L /* Connection lost or closed
/* due to a network error.

/*
 * ATM Transaction Type Codes
 */
#define OGP_ATM_BALANCE_INQUIRY 0x00000000L /* Perform balance inquiry */
#define OGP_ATM_DEPOSIT 0x00000001L /* Deposit money into acct */
#define OGP_ATM_WITHDRAWAL 0x00000002L /* Withdraw money from acct */
#define OGP_ATM_XFER_FROM_OTHER 0x00000003L /* Transfer money from acct */
#define OGP_ATM_XFER_TO_OTHER 0x00000004L /* Transfer money to acct

/*
 * Player Rating Category Codes
 */
#define OGP_PRATS_CATEGORY_SLOT 0x00000001L

/*
 * Miscellaneous definitions
 */
#define OGP_TP_SERVER_PORT 12100 /* Default TP server port */
#define OGP_TP_SERVER_PORT_NAME "ogp_txsvc" /* TP server port name */
#define OGP_START_FRAME_WORD 0x55555555L /* Start of message marker */
#define OGP_END_FRAME_WORD 0xAAAAAAAAAL /* End of message marker

#define OGP_MAX_MSG_SIZE 1024 /* Max msg size is 1024 bytes */
#define OGP_MAX_ORDER 23 /* Max. # of items that can be
/* purchased in one message

#define OGP_MAX_GAME_STATS 99 /* Max. # of game stats that can
/* be reported in one message

#endif /* OGP_H_INCLUDED */
```