[MS-MGSO]:
Multiplayer Games System Overview

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

This document provides an overview of the Multiplayer Games System Overview Protocol Family. It is intended for use in conjunction with the Microsoft Protocol Technical Documents, publicly available

standard specifications, network programming art, and Microsoft Windows distributed systems concepts. It assumes that the reader is either familiar with the aforementioned material or has immediate access to it.

A Protocol Family System Document does not require the use of Microsoft programming tools or programming environments in order to implement the Protocols in the System. Developers who have access to Microsoft programming tools and environments are free to take advantage of them.

Abstract

The DirectPlay System is designed to transport game and user data to support multiplayer gaming scenarios. The protocols in this system provide game session management as well as functionality for controlling options for sending data and voice. Control options for data include reliability, guaranteeing data delivery sequencing, and coalescence of packets. The DirectPlay System also provides functions for using network address translation (NAT).

This document describes the intended functionality of the DirectPlay System and how it interacts with applications. The DirectPlay System supports multiple protocols for multiplayer gaming. This document lists those supported protocols and how they interact in a combined system.

Revision Summary

Date	Revision History	Revision Class	Comments
02/27/2009	0.1	Major	First Release.
04/10/2009	1.0	Major	Updated and revised the technical content.
05/22/2009	2.0	Major	Updated and revised the technical content.
07/02/2009	3.0	Major	Updated and revised the technical content.
08/14/2009	4.0	Major	Updated and revised the technical content.
09/25/2009	5.0	Major	Updated and revised the technical content.
11/06/2009	5.0.1	Editorial	Revised and edited the technical content.
12/18/2009	5.0.2	Editorial	Revised and edited the technical content.
01/29/2010	6.0	Major	Updated and revised the technical content.
03/12/2010	6.0.1	Editorial	Revised and edited the technical content.
04/23/2010	6.0.2	Editorial	Revised and edited the technical content.
06/04/2010	6.0.3	Editorial	Revised and edited the technical content.
07/16/2010	6.0.3	No change	No changes to the meaning, language, or formatting of the technical content.
08/27/2010	6.1	Minor	Clarified the meaning of the technical content.
10/08/2010	6.1	No change	No changes to the meaning, language, or formatting of

Date	Revision History	Revision Class	Comments
			the technical content.
11/19/2010	6.1	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	6.1	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	6.1	No change	No changes to the meaning, language, or formatting of the technical content.
03/25/2011	6.1	No change	No changes to the meaning, language, or formatting of the technical content.
05/06/2011	6.1	No change	No changes to the meaning, language, or formatting of the technical content.
06/17/2011	6.2	Minor	Clarified the meaning of the technical content.

Contents

1 Introduction

This Protocol Family System Document (PFSD) is primarily intended to cover the Protocol Family as a whole. In conjunction with Member Protocol Technical Documents (TDs), which are intended to cover Member Protocols, it presents the rules for information exchange relevant to those Member Protocols and the Protocol Family that are used to interoperate or communicate with a Microsoft Windows® operating system in its various environments.

The Multiplayer Games System Overview describes the components necessary to implement support for a multi-**player game** using the **DirectPlay 4** and **DirectPlay 8** systems in both client/server (DirectPlay 8 only) and **peer-to-peer mode**s. For clarity, the term **DirectPlay System** will be used to refer to features common to both DirectPlay 4 and DirectPlay 8. The functionalities supported include the following:

- Game lobby support -- The game lobby services allow users to see available games on a game **host**, make configuration changes to their game properties, and join a game.
- Game state information -- Game status messages are typically provided to inform users of game server events, such as when another player joins the game.
- Side channels between game users -- Side channels provide some communication capabilities between players, usually text chat, for team coordination and taunting of opponents.
- Reliable transport -- Reliable transport provides the reliable transfer of packets containing game and side channel data.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

client
DirectPlay
DirectPlay 4
DirectPlay 4 protocol
DirectPlay 8
DirectPlay 8 protocol
DirectPlay Client
DirectPlay Host
DirectPlay Peer
DirectPlay protocol
DirectPlay Server
DirectX
game
host
host migration
local area network (LAN)
name table
network address translation (NAT)
peer
peer-to-peer mode
player
server (3)
service provider

Transmission Control Protocol (TCP)
Universal Plug and Play (UPnP)
User Datagram Protocol (UDP)

The following terms are defined in [MS-DPDX]:

data frame (DFRAME)
game session
group
instance
payload

The following terms are defined in [MC-DPL4CS]:

player ID
system player

The following terms are defined in [MC-DPL4R]:

throttling

The following terms are specific to this document:

DirectPlay 4 System: An application that implements the IDirectPlay 4 programming interface and provides peer-to-peer session-layer services to applications for multiplayer gaming.

DirectPlay 8 System: An application that implements the IDirectPlay 8 programming interface and provides client/server and peer-to-peer session-layer services to applications for multiplayer gaming.

DirectPlay System: An application that implements the IDirectPlay 4 or IDirectPlay 8 programming interface and provides client/server (DirectPlay 8 only) and peer-to-peer session-layer services to applications for multiplayer gaming.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as described in [RFC2119]. Note that in [RFC2119] terms, most of these specifications should be imperative, to ensure interoperability. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

Any specification that does not explicitly use one of these terms is mandatory, exactly as if it used MUST.

1.2 References

This section contains normative and informative references relevant to the **DirectPlay** System.

References to Microsoft Open Specification documents do not include a publishing year because links are to the latest version of the documents, which are updated frequently. References to other documents include a publishing year when one is available.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[MC-DPL4CS] Microsoft Corporation, "DirectPlay 4 Protocol: Core and Service Providers Specification".

[MC-DPL4R] Microsoft Corporation, "DirectPlay 4 Protocol: Reliable Specification".

[MC-DPL8CS] Microsoft Corporation, "DirectPlay 8 Protocol: Core and Service Providers Specification".

[MC-DPL8R] Microsoft Corporation, "DirectPlay 8 Protocol: Reliable Specification".

[MC-DPLHP] Microsoft Corporation, "DirectPlay 8 Protocol: Host and Port Enumeration Specification".

[MC-DPLNAT] Microsoft Corporation, "DirectPlay 8 Protocol: NAT Locator Specification".

[MC-DPLVP] Microsoft Corporation, "DirectPlay Voice Protocol Specification".

[MS-DPDX] Microsoft Corporation, "DirectPlay DXDiag Usage Protocol Specification".

[MS-NLMP] Microsoft Corporation, "NT LAN Manager (NTLM) Authentication Protocol Specification".

[RFC768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980, http://www.ietf.org/rfc/rfc768.txt

[RFC791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981, http://www.ietf.org/rfc/rfc791.txt

[RFC793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981, http://www.ietf.org/rfc/rfc0793.txt

[RFC1132] McLaughlin, L., III, "A Standard for the Transmission of 802.2 Packets over IPX Networks", STD 49, RFC 1132, November 1989, http://www.ietf.org/rfc/rfc1132.txt

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

[UPNPWANIP] UPnP Forum, "WANIPConnection:1", November 2001, http://www.upnp.org/standardizeddcps/documents/UPnP_IGD_WANIPConnection%201.0.pdf

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary".

2 Overview

Section 1, "Introduction", describes this Protocol Family System Document. This section introduces the system that is being documented.

2.1 System Summary

The DirectPlay System was developed for multiplayer gaming over IP/IPX networks, serial connections, and modems. The DirectPlay interfaces were released with the **DirectX** Software Development Kit (SDK) and the DirectX End User Runtime.<1>

The DirectPlay System consists of protocols that enable multiplayer game functionality: the DirectPlay System Protocols and the DirectPlay Voice Protocol. There are two versions of the DirectPlay System protocols, the **DirectPlay 4 System** protocols and the **DirectPlay 8 System** protocols. The DirectPlay 4 System protocols and DirectPlay 8 System protocols are very similar conceptually. In fact, applications would not normally implement both DirectPlay 4 System and DirectPlay 8 System protocols due to the overlap in functionality. This document describes the relationships among each system's protocols, as well as how the **DirectPlay 8 protocols** use host port enumeration and **network address translation (NAT)** resolver.

The DirectPlay 4 System Protocol comprises the protocols specified in [MC-DPL4CS] and optionally [MC-DPL4R], and the DirectPlay 8 System Protocol comprises protocols [MC-DPL8CS] and [MC-DPL8R]. The use of the DirectPlay 8 System Protocol by the DxDiag system application in Microsoft Windows® is covered in [MS-DPDX]. The DirectPlay 8 System MAY also optionally use the DirectPlay 8 Protocol: Host and Port Enumeration Specification [MC-DPLHP] and DirectPlay 8 Protocol: NAT Locator Specification MC-DPLNAT. Both DirectPlay System Protocols support optional use of the DirectPlay Voice Protocol [MC-DPLVP].<2>

There are two main components of each version of the DirectPlay System Protocol: Core and Service Providers for managing the **game session**, including adding and removing players, and Reliable Data Transport to implement reliable and unreliable delivery of messages.

The DirectPlay Core and Service Providers protocols enable **DirectPlay Clients**/**Peers** within a DirectPlay game session to communicate multiplayer game session information. The exchange is coordinated by either the **server** (DirectPlay 8 only) or a host **peer**. The protocol depends on the underlying reliable transport mechanism to handle connectivity and transport of messages. Most of the implementations of DirectPlay used by retail games run in peer-to-peer mode; however, DirectPlay 8 also includes the ability for games to run in **client**/server mode.

The DirectPlay reliable protocols, [MC-DPL4R] and [MC-DPL8R], describe the reliable transport mechanism. The reliable transport mechanism combines the reliable and unreliable delivery of messages, as well as the sequential and nonsequential delivery of messages, into a single channel of data. For [MC-DPL4R], the reliable protocol optionally may also provide throttling. Games have different requirements for different types of messages. For example, a message related to a player position doesn't always need to be sent reliably, and if a packet is lost it is not necessarily critical to the outcome of the game; however a message related to player scores needs to be sent reliably; all players would want to have accurate knowledge of the score.

The DirectPlay Voice Protocol enables voice communication within an application game session. The exchange is coordinated by a voice server and works independently of the version of the DirectPlay System in use. The DirectPlay Voice Protocol depends on the underlying game session to handle connectivity and transport between the voice clients and the voice server.

The **DirectPlay 8 Host** and Port Enumeration Protocol is a simple query/response protocol that allows clients to find **hosts** offering **game sessions** that are willing to accept the client as an

additional participant. The criteria that **hosts** use when determining whether to offer the **game session** or that clients use when selecting a **game session** given multiple responses are left to the implementation and the end user.

Three member protocols, [MC-DPL8CS], [MC-DPL4CS], and [MC-DPLVP], are concerned with managing multiple players simultaneously within a single **game session** or voice session instance. These protocols also implement optional **host migration**, so that the **game session** or voice session management can continue even if the original host leaves or is disconnected. The algorithms used by each protocol to select a new host are similar, but not identical. They are also logically independent; the **DirectPlay 4 Protocol**: Core and Service Providers is not expected to be implemented in an application that is also using DirectPlay 8 Protocol: Core and Service Providers, and there is no requirement that the DirectPlay Voice Protocol voice server be co-located with a Core and Service Providers host, or even that the Core and Service Providers host participate in the voice session at all.

DirectPlay has few interactions with other Windows protocols or services. DirectPlay 4 can use NTLM for player authentication.

The DirectPlay System is officially deprecated.<3>

2.2 List of Member Protocols

The DirectPlay System implements the protocols specified in [MS-DPDX], [MC-DPL4CS], [MC-DPL4R], [MC-DPL8CS], [MC-DPL8R], [MC-DPLHP], [MC-DPLNAT], and [MC-DPLVP] as follows:

DirectPlay 4 Protocol, as specified in [MC-DPL4CS], [MC-DPL4R], and [MS-DPDX]. These technical documents describe messaging and specify transport-level message delivery capabilities as well as application-level support for session setup, user and **group** management, data delivery, host migration, and a diagnostic utility.

DirectPlay 8 Protocol, as specified in [MC-DPL8CS], [MC-DPL8R], [MC-DPLHP], [MC-DPLNAT], and [MS-DPDX]. These technical documents describe messaging and specify transport-level message delivery capabilities as well as application-level support for session setup, user and group management, data delivery, host migration, improved (NAT) traversal, improved session discovery, and a diagnostic utility.

DirectPlay Voice Protocol, as specified in [MC-DPLVP]. This protocol specifies a session layer for multipoint audio conferencing that can use DirectPlay 4 or DirectPlay 8 as the transport.

2.3 Relevant Standards

The system uses the standards listed below to allow interoperability with other external systems.

Transmission Control Protocol (TCP), as specified in [RFC793]. This standard can be used to provide transport for DirectPlay 4.

User Datagram Protocol (UDP), as specified in [RFC768]. This standard can be used to provide transport for DirectPlay 4, DirectPlay 8, and DirectPlay Voice.

Internet Protocol (IP), as specified in [RFC791]. This standard can be used to provide transport for DirectPlay 4, DirectPlay 8, and DirectPlay Voice.

Universal Plug and Play (UPnP) Internet Gateway Device (IGD) V 1.0, as specified in [UPNPWANIP]. This standard supports communication with NATs for DirectPlay 8.

Internet Package Exchange (IPX), as specified in [RFC1132]. This standard can be used to provide transport for DirectPlay 4 and DirectPlay 8.

Recommended Standard 232 (RS-232), as specified in multiple sources. This standard can be used to provide serial and modem transport for DirectPlay 4 and DirectPlay 8.

NT LAN Manager (NTLM), as specified in [MS-NLMP]. This standard can provide optional authentication for DirectPlay 4.

3 Foundation

This section describes the theoretical and practical information needed to understand this document and this system.

3.1 Background Knowledge and System-Specific Concepts

This section summarizes background knowledge required to understand this document. It is assumed that the reader of this document has the following background knowledge:

- General computer science theory and some familiarity with network programming concepts
- Familiarity with peer-to-peer networking concepts
- Familiarity with client/server technology
- Familiarity with block protocols

There are no system-specific concepts.

3.2 System Purposes

The DirectPlay System exists in an attempt to make network gaming development easier and more robust. The mission is to enable multiplayer gaming with a variety of network transports in a variety of network topologies.

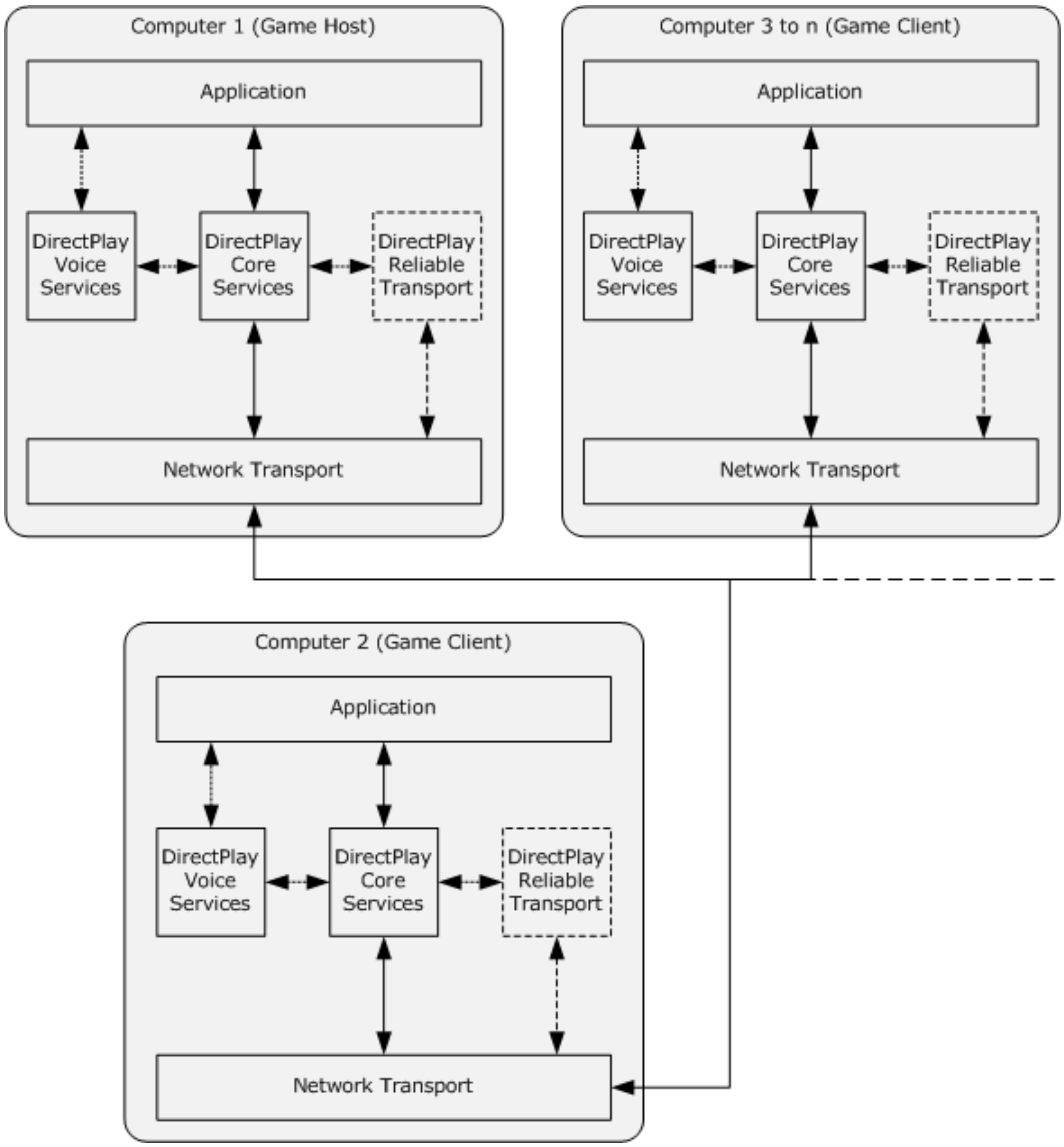

Figure 1: DirectPlay 4 System component relationship diagram

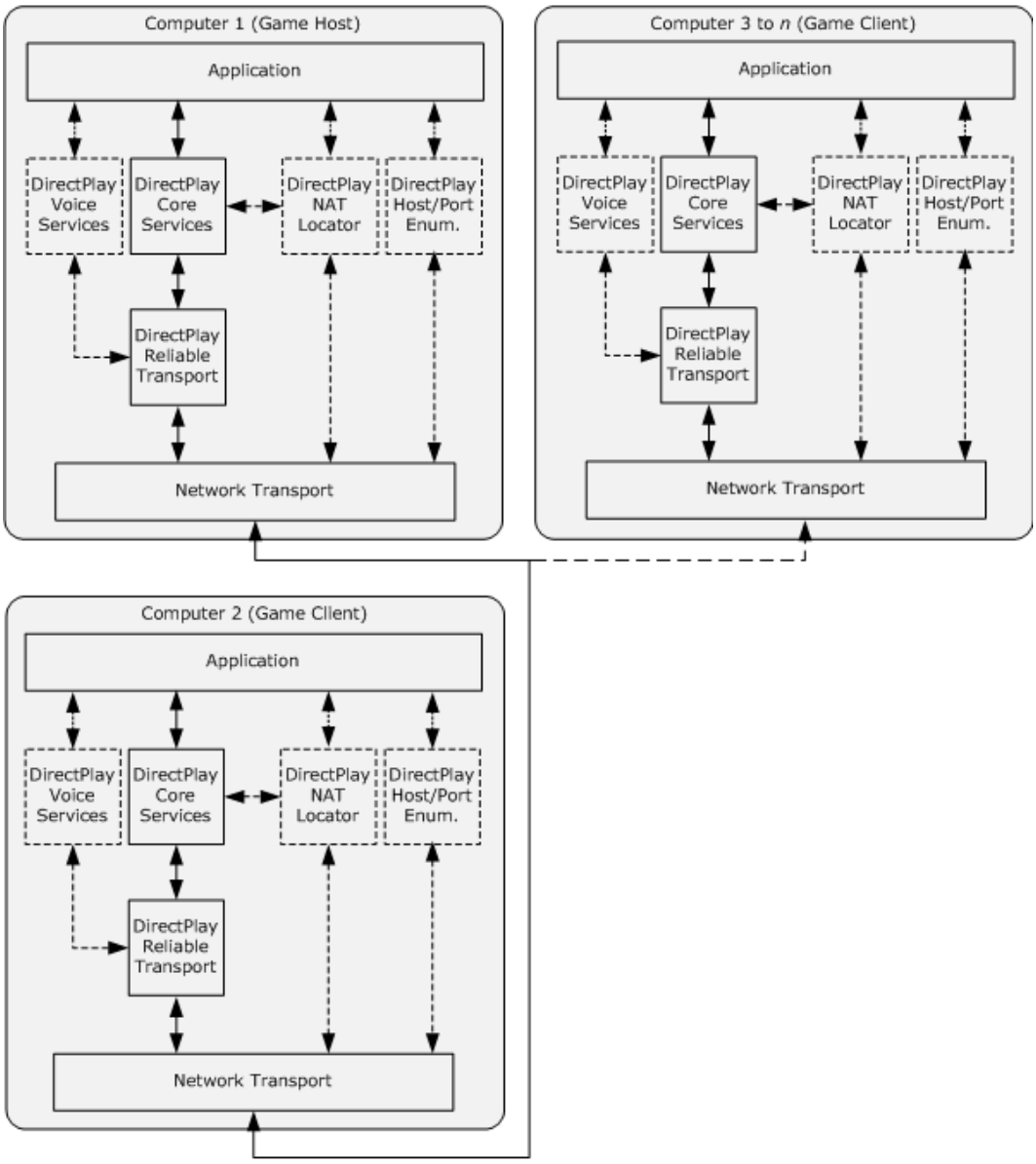

Figure 2: DirectPlay 8 System component relationship diagram

3.3 System Use Cases

This section specifies the major use cases of the DirectPlay System and the rationale for their usage. The DirectPlay System is designed to transport game and user data for multiplayer games scenarios.

3.3.1 Stakeholders and Interests Summary

The stakeholders in the DirectPlay System are game designers and architects, developers, testers, and users.

3.3.1.1 Game Designers and Architects

An architect is responsible for the overall design of the implementation of a game while managing the technical risks associated with it.

The DirectPlay System is one of many environments that enable network play. The architect and game designer need to decide many aspects of the game, such as number of players, game play modes that are supported, network topology used in the game, features that are part of the networking solution such as leaderboards, and voice support. The architect also needs to design messages, data, and frequency of data used by the game.

An architect can use the DirectPlay System as an element of the game that he is designing to offer enumeration, connectivity, and communication between the players involved in the game. Section 3.3.3 goes into greater detail on each of these concepts.

The system described here offers the architect the following advantages:

- Support for peer-to-peer or client/server topologies.
- NAT traversal help (DirectPlay 8 System only).
- Reliable/unreliable sending of data.
- Enumeration of players.
- Connection to other players.
- Host migration.

3.3.1.2 Developers

Developers have a more feature-centric view of the system and are likely to focus on one or more individual features, such as host migration, creating a session, joining a session, leaving a session, sequential data transmission, nonsequential data transmission, reliable data transmission, creating and dismantling groups, and NAT traversal.

Generally, on a game development team, there is one network developer. The value offered by the DirectPlay System is in providing prescribed message handling sequences for many common situations and race conditions that the developer would otherwise have to design on his or her own. For example, [MC-DPL8CS] defines a single central owner for shared state, the host, who assigns sequence identifiers to "nametable" operations like players joining or leaving to ensure that they are performed in order, as described in [MC-DPL8CS] section 2.2.6. If two players appear to join simultaneously, one of the players will actually be first by virtue of assigning sequence identifiers, and all existing participants will then agree on this order. Another example is that [MC-DPL8CS] section 1.3.5 prescribes an integrity checking procedure to recover from peers having disparate views of the list of participants after the connection between two peers is broken. As a result of these behaviors, successful implementation of the DirectPlay System implies proper handling of many typical scenarios that are encountered as part of game development.

3.3.1.3 Testers and Quality Assurance (QA) Personnel

Testers and quality assurance (QA) personnel have a feature-centric view of the system similar to developers, but are focused at a functional level rather than at a development level. QA personnel are likely to focus on one or more individual features, such as creating a session, joining a session, leaving a session, and host migration.

The DirectPlay System allows testers and quality assurance personnel to focus more of their energy on core game play and less on the transport and race conditions.

3.3.1.4 Users

Users are the people who play the games.

3.3.2 Supporting Actors and System Interests Summary

The DirectPlay System does not have any Supporting Actors.

3.3.3 Use Case Diagrams

This section presents the DirectPlay System use case diagram and describes elements in the diagram.

There are three actors in this system:

- User: This is the user of a specific game application. The user decides what role the game application, and thus the DirectPlay System, will play.

  The game application can run in two modes: a Host Entity and a Client Entity.

- Host Entity: A Host Entity is established by a user who wants to begin a multiplayer game session and be joined by other Client Entities.

- Client Entity: A Client Entity is established by a user who wants to join a multiplayer game Host Entity.

The figure below presents an overview of the use cases for the DirectPlay System: host a DirectPlay game and join a DirectPlay game. Enumeration, connectivity, and communication are subtasks of the primary use cases.

Enumeration

Enumeration is the initial action for the DirectPlay System. The Host Entity (game) will initialize the DirectPlay System and indicate to the system that it wishes to become the host for a multiplayer game. The Client Entity (game) will also initialize the DirectPlay System; however, it will indicate to the system that it wishes to discover and join a game that is already being hosted.

The DirectPlay Client/Peer will begin inquiring for ongoing games that are available to join. Upon receipt of an inquiry, the **DirectPlay Server**/**Host** will respond to the DirectPlay Client/Peer with the details on the Host Entity (game) to be used in Connectivity.

Details of the messages passed between Host and Client Entities are described in [MC-DPL4CS] sections 3.1.4.1 and 3.2.5.3, and in [MC-DPLHP] section 2.2.

Connectivity

Connectivity allows for a Host Entity and Client Entity to establish a connection between each other. Connectivity typically begins with the DirectPlay Client/Peer selecting a result from Enumeration, and issuing a request to a DirectPlay Server/Host to begin communication. The DirectPlay Server/Host will respond with acceptance or denial of connection to the game. If the Client Entity is accepted, the DirectPlay Server/Host will then respond with game information to the Client Entity and establish connectivity. For further information, see DirectPlay 4 Protocol: Core and Service Providers Specification [MC-DPL4CS], Section 3.2 and DirectPlay 8 Protocol: Core and Service Providers Specification [MC-DPL8CS], Section 3.1.

After establishing the connection (communication may or may not have already occurred), it is up to the Host Entity and to the DirectPlay Server/Host to continue managing the connections with the DirectPlay Client/Peer, making alterations to the connection as needed. This MAY include disconnecting clients, if necessary.

Communication

After Enumeration and establishing Connectivity, the Host Entity and Client Entity can begin Communication. Communication MAY also occur between a new Client Entity and other Client Entities that have previously joined successfully. Communications include both the game play and the session management messages. Communication of game data is very specific to the implementation of the game. Communication of session information involves common operations such as Client Entities joining or leaving the multiplayer game session, as defined in the Core and Service Providers Protocols [MC-DPL4CS] and [MC-DPL8CS], but the exact subset or sequences performed are still specific to the game implementation and typically result from actions by users.

Game data messages can be sent reliably or unreliably, also known as "best effort", (see [MC-DPL4R] section 2 and [MC-DPL8R] section 2.2.2).Other properties can also be set. For example, for coalescence the DirectPlay System will attempt to put as many messages as possible in a packet (see [MC-DPL8R] Section 2.2.3). Finally, both DirectPlay 4 and DirectPlay 8 offer the ability to send a message larger than the MTU, which is broken down into multiple packets and reassembled on the receiving system (see [MC-DPL4R] Section 3.1.5.1, and [MC-DPL8R] Section 3.1.5.2.6).

Figure 3: DirectPlay System use cases

3.3.4 Use Case Descriptions

3.3.4.1 Host a DirectPlay Game—Host Entity

Goal: The goal of the use case is to host a DirectPlay game session.

Context of use: This use case is initiated when the user selects the Host Entity role, which the game application communicates to the DirectPlay System.

Direct Actor: The direct actor of this use case is the Host Entity.

Primary Actor: The primary actor is the user.

Supporting Actors: None.

Stakeholders and Interests:

- The user who wants to host a game.
- Other users who want to connect to a game.
- The Host Entity initializes the DirectPlay System in order to host the game.
- The Client Entity initializes the DirectPlay System in order to join a game.

Preconditions:

- The user must install the game application that uses DirectPlay.
- Network connectivity needs to be available.

Minimal Guarantees:

- The Host Entity can initialize the DirectPlay System.
- The Client Entity can initialize the DirectPlay System.

Success Guarantee:

- The Host Entity can initialize the DirectPlay System.
- The Client Entity can initialize the DirectPlay System.
- The Host Entity successfully hosts a game session.
- The Client Entity successfully joins a game hosting session.
- All available games are enumerated.
- Host Entity and Client Entity are connected.
- Game data is exchanged.

Trigger:

- The user triggers this use case.

Main Success Scenario:

1. A user selects the option to host the game session.
2. The Host Entity initializes the DirectPlay System.
3. A Client Entity requests enumeration of available games using DirectPlay.
4. Connectivity is established between Host and Client.
5. Game data is exchanged between Host and Client Entities.

Extensions: None.

3.3.4.2 Join a DirectPlay Game—Client Entity

Goal: The goal of the use case is to join a DirectPlay game.

Context of use: This use case is initiated when the user selects the Client Entity role, which the game application communicates to the DirectPlay System.

Direct Actor: The direct actor of this use case is a Client Entity.

Primary Actor: The primary actor is the user.

Supporting Actors: None.

Stakeholders and Interests:

- A user who wants to host a game.
- A user who wants to connect to a game.
- The Host Entity initializes the DirectPlay System in order to host the game.
- The Client Entity initializes the DirectPlay System in order to join a game.

Preconditions:

- The user must install the game application that supports DirectPlay.
- Network connectivity needs to be available.

Minimal Guarantees:

- The Host Entity can initialize the DirectPlay System.
- The Client Entity can initialize the DirectPlay System.

Success Guarantee:

- The Host Entity can initialize the DirectPlay System.
- The Client Entity can initialize the DirectPlay System.
- The Host Entity successfully hosts a game session.
- The Client Entity successfully joins a game hosting session.
- All available games are enumerated.
- User selects an available game session.
- Host Entity and Client Entity are connected.
- Game data is exchanged.

Trigger:

- The user triggers this use case.

Main Success Scenario:

1. A user selects the option to host the game session.

2. The Host Entity initializes the DirectPlay System.
3. The user selects the option to join a game session.
4. A Client Entity requests enumeration of available games using DirectPlay.
5. The user selects a game session.
6. Connectivity is established between Host and Client Entities.
7. Game data is exchanged between Host and Client Entities.

Extensions: None.

3.3.4.3 Migrate a Game or Voice Session to a New Host—Host Entity

Goal: The goal of the use case is to enable the host migration of a DirectPlay game or voice session from an existing Host Entity to a new Host Entity, which was formerly a Client Entity.

Context of use: This use case is initiated when the Host Entity user chooses to leave the DirectPlay Session, which the game or voice application communicates to the DirectPlay System.

Direct Actor: The direct actor of this use case is a Host Entity.

Primary Actor: The primary actor is the user.

Supporting Actors: At least one Client Entity.

Stakeholders and Interests:

- A user who wants to stop hosting a game or voice session.
- A user who is able to assume hosting a game or voice session.
- The Host Entity communicates a change in status to the DirectPlay System in order to stop hosting the game or voice session.

Preconditions:

- Both users must install the game or voice application that supports DirectPlay.
- Network connectivity needs to be available.
- Both users are in the same game session or voice session.

Minimal Guarantees:

- The Host Entity can initialize the DirectPlay System.
- The Client Entity can initialize the DirectPlay System.

Success Guarantee:

- The Host Entity can initialize the DirectPlay System.
- The Client Entity can initialize the DirectPlay System.
- The Host Entity successfully hosts a game session or a voice session.

- The Client Entity successfully joins a game or voice hosting session.
- All available games or voice sessions are enumerated.
- The Host Entity and Client Entity are connected to the same game session or voice session.
- The initial Host Entity communicates a change in status to the DirectPlay System in order to stop hosting the game or voice session.
- The successor Host Entity, as determined by the host migration algorithms specified in [MC-DPL4CS] section 3.1.7.1, [MC-DPL8CS] section 1.3.6, and [MC-DPLVP] section 1.3.3.1, communicates intent to assume hosting to the DirectPlay System.
- The hosting role is successfully assumed by the new Host Entity.
- Game data is exchanged.

Trigger:

- The initial hosting user triggers this use case.

Main Success Scenario:

1. A user deselects the option to host the game session or voice session.
2. The Host Entity communicates the desired change in status to the DirectPlay System.
3. A Client Entity assumes the Host Entity role.
4. Game data is exchanged between Host and Client Entities.

Variation:

- Direct Actor: Client Entity.

Extensions: None.

4 System Context

This section describes the relationship between this system and its environment.

4.1 System Environment

The DirectPlay System was intended to be used by game applications that need a communication mechanism to enable multiplayer gaming. In order to facilitate this communication, the DirectPlay System must have access to a transport for the delivery of its messages. The majority of game applications will choose to have the DirectPlay System use an IP network. Additionally, the DirectPlay System can also support IPX networks. In each case, the network must have visibility and connectivity to the other DirectPlay systems with which it is attempting to communicate.

Beyond IP or IPX networks mentioned above, the DirectPlay System also provides support for communication through two additional mediums. These mediums include direct serial cable connections and Modem-to-Modem communication. In both of these cases, the underlying connections must already be established when the DirectPlay System is launched.

4.2 System Assumptions and Preconditions

The system assumes that the user has launched a game application and intends to either create a new game session, search for an existing session, or join a game session that has already been located. In addition to defining the system's role in the session, the game application specifies the parameters of how one **instance** of the DirectPlay System will connect to another instance. The parameters include the network type (IP, IPX, Serial, Modem) that will be used to transport the messages between the DirectPlay Systems, the version of the DirectPlay System to be used, and the connection options. DirectPlay 4 MAY optionally call the NT LAN Manager (NTLM) Authentication Protocol [MS-NLMP].

4.3 System Relationships

This section describes the relationships between the system and external components, system dependencies, and other systems influenced by this system.

4.3.1 Black Box Relationship Diagram

The following diagram shows how a given instance of the DirectPlay System relates to other protocols and components.

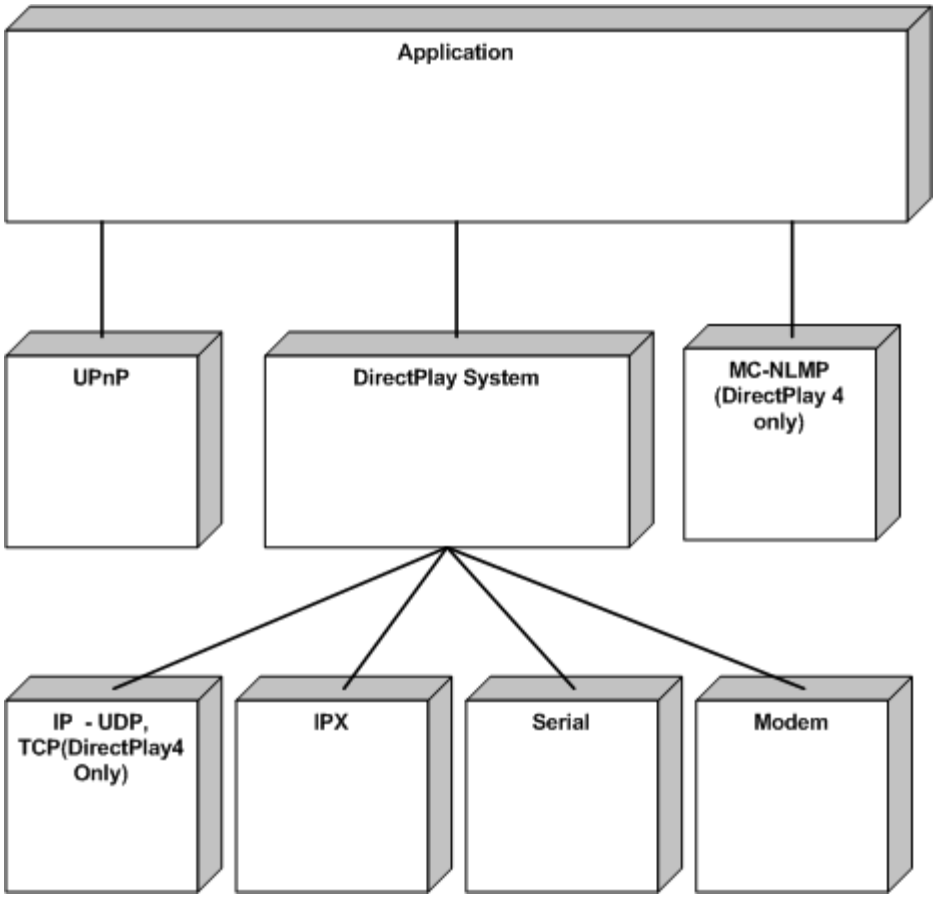

Figure 4: DirectPlay System relationship to external components

The game application uses interfaces provided by the DirectPlay core and **service provider**s to invoke system functionality. The types of operations available and their messages are described in [MC-DPL4CS] and [MC-DPL8CS].

If the game application has chosen to use an IP network, the DirectPlay System's operation and application-specified **payload**s are transported using **UDP**. However, in a DirectPlay 4 System, some of the messages are also transported using **Transmission Control Protocol (TCP)** payloads. All messages will exit an abstract DirectPlay System instance by using one of these two protocols.

If the game application has chosen to use an IPX network, the DirectPlay System's operation and application-specified payloads are transported using IPX datagrams.

If the game application has chosen to use a serial or modem connection, the DirectPlay System's operation and application-specified payloads are transported using binary payloads. The serial or modem connection should already be established prior to communication.

An application using the DirectPlay System MAY also choose to invoke Universal Plug and Play Internet Gateway Device control functionality to create port mappings. This is not a requirement to successfully participate in the DirectPlay System; however, it is mentioned here because peer-to-peer mode communication protocols designed for consumer game applications such as DirectPlay may benefit from the increased Network Address Translation compatibility that such mappings provide. Because the **DirectPlay protocol**s allow the application to select which UDP and TCP ports

to use, the actual port numbers to map using **Universal Plug and Play (UPnP)** are not prescribed here.

A game application using the DirectPlay 4 System may also choose to create a secure session that requires authentication. The application and environment MUST be configured to use the NT **LAN** Manager protocol; the DirectPlay 4 System will then invoke the authentication sequence to validate players that join the game session as described in [MC-DPL4CS]. This authentication functionality is not available when using the DirectPlay 8 System.

The following diagram illustrates how an application perceives concepts exposed by the DirectPlay System, specifically the core and service provider protocols described in [MC-DPL4CS] and [MC-DPL8CS].

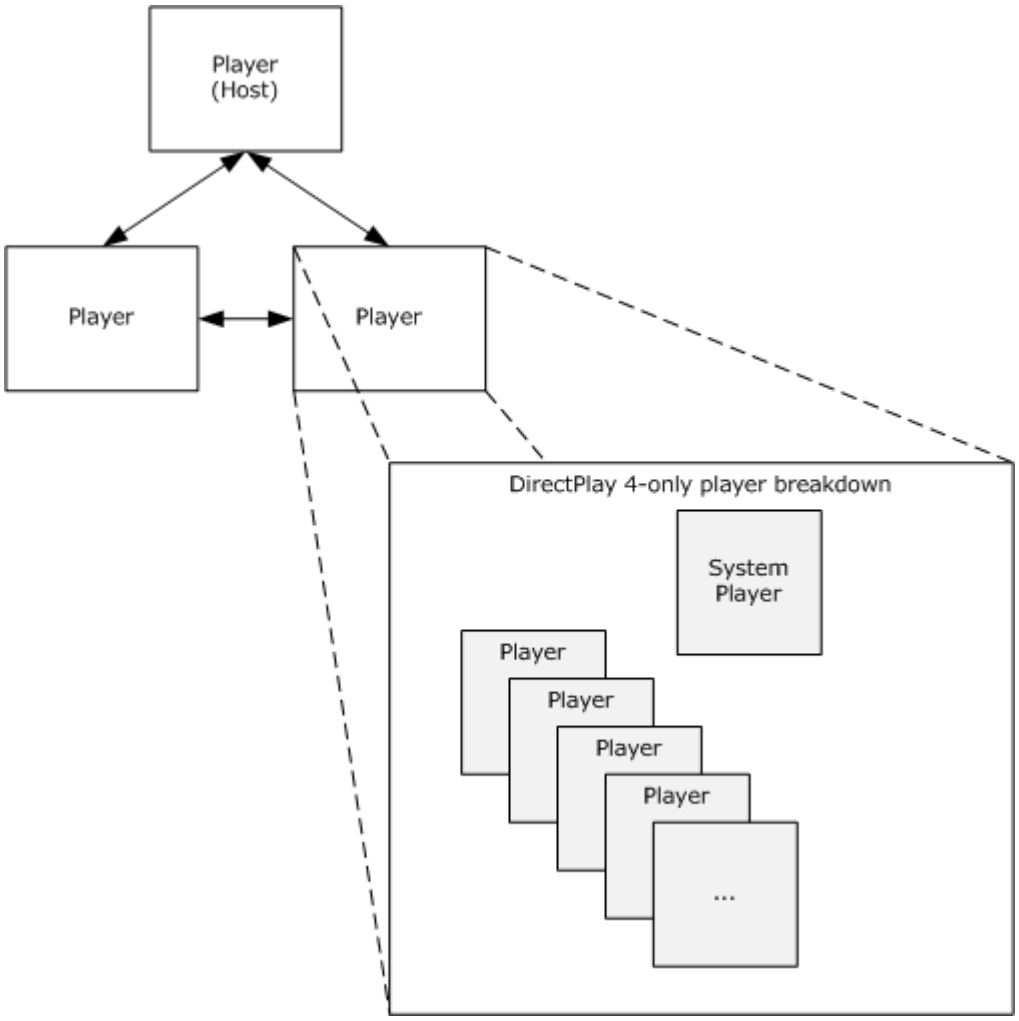

Figure 5: Virtual components in a DirectPlay System game session

Each player square represents a peer application instance participating in the DirectPlay System game session. One, and only one, of these players is designated as the host. The host has certain responsibilities, such as managing the **name table**, as outlined in the core and service provider protocol documents [MC-DPL4CS] and [MC-DPL8CS]. There may be one or more peers participating in a game session at any time.

An application using the DirectPlay 4 System breaks down each peer application instance into subcomponents: a **system player** and zero or more nonsystem players. The system player is not typically exposed to an application by a DirectPlay implementation.

4.3.2 System Dependencies

The DirectPlay System depends on the configurations and requests that are defined by the game application that is using the system.

These dependencies include:

- If the game application selects the IP network to be used by the DirectPlay System, the machine MUST have an IP address available.
- If the game application has chosen the IP network to be used by the DirectPlay System, the UDP protocol MUST be available.
- If the game application has chosen the IPX network to be used by the DirectPlay System, the machine MUST have an IPX address available.
- If the game application has chosen to use a serial connection by the DirectPlay System, the machine MUST have an external operating system component (eg. Direct Cable Connection) capable of sending and receiving binary data over the connection.
- If the game application has chosen to use a serial or modem connection by the DirectPlay System, the participants in the session SHOULD have already established a connection to each other prior to invoking the DirectPlay System.

The DirectPlay 4 System has the following additional dependencies:

- If the game application has chosen the IP network to be used by the DirectPlay System, the TCP protocol MUST be available.
- If the game application has chosen to use the secure session option, valid NTLM accounts MUST be available to support authentication.

4.3.3 System Influences

If the game application has chosen to use an IP network for communication, the DirectPlay System will attempt to discover external networking devices that it may need to traverse as defined in [MC-DPLNAT]. Upon discovery of these devices, the DirectPlay System may hand up to the game application additional connectivity options that the application may choose. By handing up these additional connectivity options to the game application, the DirectPlay System confirms its ability to establish connections through these additional paths.

By default, the DirectPlay System will attempt to claim specified networking ports on a system. Except for the case of [MC-DPLHP], the DirectPlay System is able to move within a range of options to select a port that is available. This is done to both avoid conflict, and also to allow for more than one DirectPlay System instance per machine.

4.4 System Applicability

As mentioned in Section 2, the DirectPlay System was designed to be used for multiplayer game applications. The two gaming scenarios being supported include peer-to-peer gaming and client/server gaming. A peer-to-peer system is ideal when the game application only needs to connect a limited number of peers together. Because each peer has a direct connection to each

other peer in the session, latency is minimized and the messages are sent directly to the targeted peer. However, due to the number of connections associated with a peer-to-peer system, there is a lot of overhead in coordination of the peers and instances as players are added and dropped within the session, thus the need to limit the number of peers.

If the game application will have a large number of players within the game, the application may choose the client/server scenario. This will simplify connectivity because there is only one host for the session. The clients are able to query the host on a well-known port as specified in [MC-DPLHP], and with the response from the server, are able to connect. Also, because each client has only one connection to the server, the system can support a much larger set of DirectPlay System instances than a peer-to-peer system. The main disadvantage of a client/server system is that the clients are unable to talk directly to each other, which will increase latency.

Beyond the two scenarios mentioned above, a game application may also chose to use the DirectPlay System to leverage the connectivity improvements as defined in [MC-DPLNAT]. These will assist in traversing a variety of networking hardware including NATs, routers, and firewalls.

If a game application also requested voice communication between its instances, the DirectPlay System supports this scenario as defined by [MC-DPLVP]. The DirectPlay System provides a variety of voice encoders that can be chosen by the game application based on the needed quality and the allowable bandwidth as defined by the game application.

4.5 System Versioning and Capability Negotiation

The DirectPlay 4 and DirectPlay 8 systems are not interoperable. There is no defined versioning or capability negotiation that will enable DirectPlay 4 and DirectPlay 8 to interoperate. Each protocol in the system treats unexpected messages, such as those from the counterpart system, as invalid and reacts as described in the specific protocol specifications.

The DirectPlay 4 and DirectPlay 8 systems SHOULD have their component protocols implemented together using the highest version available for each, but this is not a requirement. There is no negotiation performed among the protocols at runtime to determine their relative versions or capabilities.

4.6 System Vendor-Extensible Fields

There are no fields designed for vendors to extend the aggregate DirectPlay System. However, the DirectPlay Systems themselves are generic transports for external gaming applications; therefore, the payloads transported, such as game-specific messages, or game-specific buffers of data to associate with a Player, could be used for vendor-extensibility.

For the DirectPlay 4 System, these include, but are not limited to:

- [MC-DPL4CS] section 2.2.2 DPLAYI_PACKEDPLAYER **PlayerData**, **ShortName**, and **LongName** fields
- [MC-DPL4CS] section 2.2.19 DPSP_MSG_CHAT **ChatMessage** field
- [MC-DPL4CS] section 2.2.46 DPSM_MSG_PLAYERMESSAGE **PlayerMessage** field

For the DirectPlay 8 System, these include, but are not limited to:

- [MC-DPL8CS] section 2.2.1.2 DN_INTERNAL_MESSAGE_PLAYER_CONNECT_INFO_EX **connectData**, **data**, and **name** fields

- [MC-DPL8CS] section 2.2.1.4 DN_SEND_CONNECT_INFO **Data**, **Name**, **SessionName**, and **Reply** fields
- [MC-DPL8CS] section 2.2.4.1 DN_REQ_CREATE_GROUP **data** and **name** fields
- [MC-DPL8R] section 2.2.2 Data Frame (DFRAME)s **payload** field
- [MC-DPLHP] section 2.2.1 EnumQuery **ApplicationPayload** field
- [MC-DPLHP] section 2.2.2 EnumResponse **ApplicationData** field

5 System Architecture

This section describes the basic structure of the system and the interrelationships among its parts, consumers, and dependencies.

5.1 Abstract Data Model

Specific data and state elements used by the DirectPlay System are almost exclusively managed as part of the implementation of their containing member protocol. However, a few elements and concepts are shared among these protocols and should be considered holistically with the DirectPlay System.

5.1.1 Game Session Description

The foundation of the DirectPlay System is the game session. This is conceptually owned by the Core and Service Providers protocols [MC-DPL8CS] and [MC-DPL4CS], which provide explicit management transactions such as those to add or remove players to the particular instance of the game.

However, several data elements are commonly used when describing a game session either to end users or to applications so that they can make decisions about whether to join the game session. Game host or server applications that support discovery using the DirectPlay 8: Host and Port Enumeration protocol should be prepared to report these descriptive elements in the EnumResponse messages described in [MC-DPLHP] section 2.2.2 in addition to transmitting them as part of the DirectPlay 8: Core and Service Providers protocol [MC-DPL8CS] section 2.2.1.4. The shared session description elements are:

MaxPlayers: The maximum number of players who can join the game session.

CurrentPlayers: The number of players currently participating in the game session.

ApplicationInstanceGUID: A **GUID** that identifies the particular instance of the application. Each instance of a DirectPlay System application generates a new **GUID** each time it chooses to host a new game session. The **GUID** can be used to distinguish game sessions when the host creates a game session, terminates it, and then creates a different one using the same network address, for example.

ApplicationGUID: A **GUID** that distinguishes the particular DirectPlay System application from other applications that also use the DirectPlay System.

SessionName: A human-readable name for the game session. This is typically provided by the end user.

ApplicationReservedData: Any other application-specific data that helps describe the game session to current and potential participants.

Applications implementing the optional DirectPlay 8 Protocol: NAT Locator Path Test extension to DirectPlay 8 Protocol: Core and Service Providers should note that the ApplicationInstanceGUID and ApplicationGUID are also referenced in [MC-DPLNAT] section 2.2.1 as part of generating the key value.

5.1.2 Player Identification

Each participant in a DirectPlay System game session is represented by a player, as illustrated in the figure titled Virtual components in a DirectPlay System game session in section 4.3.1. Adding

players to the game session and assigning them unique identifiers is managed by the Core and Service Providers protocols [MC-DPL8CS] and [MC-DPL4CS].

However, applications may choose also to use the [MC-DPLVP] Voice protocol, which uses the available Core and Service Providers protocol players or a subset as the list of potential voice players in a voice session. The DPNID or **player ID** unique identifiers assigned by [MC-DPL8CS] section 2.2.7 or [MC-DPL4CS] section 3.2.5.4 are used directly as the DVID identifiers for a voice session participant by [MC-DPLVP]. The following figure shows a possible hierarchy of how an implementation might store player state, including its identifier, when using the DirectPlay Voice Protocol. DirectPlay 8 Protocol: Core and Service Providers player entities "1", "3", and "..." have corresponding DirectPlay Voice Protocol player entities ("1", "2", and "..." respectively). DirectPlay 8 Protocol: Core and Service Providers player "2" does not have a corresponding DirectPlay Voice Protocol player.

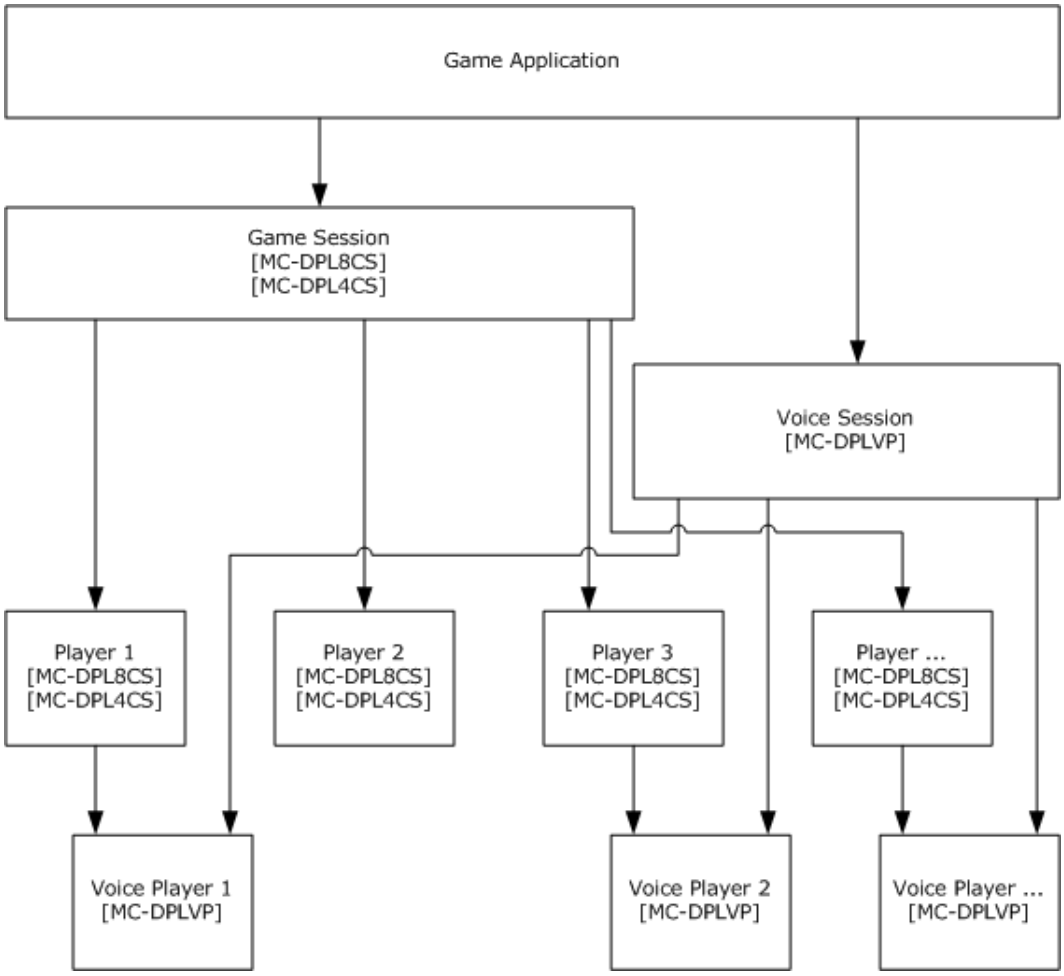

Figure 6: Example Player and Voice Player State Object Ownership Hierarchy

The DPNIDs of the two participating players are also referenced in [MC-DPLNAT] section 2.2.1 as part of generating the key value for applications implementing the optional DirectPlay 8 Protocol: NAT Locator Path Test extension to DirectPlay 8 Protocol: Core and Service Providers.

5.2 White Box Relationships

The Protocol Stack diagrams in section 6.3 illustrate the relationship of the components of the DirectPlay Systems. The game application is used as the initial entry point to the system and is defined further in section 4.3.1.

If the game application has chosen to create a DirectPlay Voice Session utilizing the DirectPlay Voice Protocol [MC-DPLVP], the DirectPlay Voice protocol in turn will pass its data onto the DirectPlay Core and Service Providers protocol.

The DirectPlay Core and Services Providers protocol is the entry point for the game application into the DirectPlay System. All application data will be passed through the DirectPlay Core and Services Providers protocol. Additionally, the DirectPlay 4 Core and Services Providers protocol may choose to leverage the DirectPlay 4 Reliable Protocol [MC-DPL4R] along with communicating directly with the chosen transport. For the DirectPlay 8 Core and Service Providers protocol, all communication is sent through the DirectPlay 8 Reliable Protocol [MC-DPL8R]. Additionally, the DirectPlay 8 Core and Service Providers protocol may also utilize the enumeration functionality of the DirectPlay 8 Host and Port Enumeration Specification [MC-DPLHP] along with the NAT capabilities defined in the DirectPlay 8 NAT Locator protocol [MC-DPLNAT].

The DirectPlay Reliable Protocols [MC-DPL4R] and [MC-DPL8R] are responsible for the actual transmission of data through the chosen transport.

5.3 Member Protocol Functional Relationships

This section provides the details on how each member protocol is used by the system to perform certain roles and the interrelationships and dependencies between the protocols.

5.3.1 Member Protocol Roles

The DirectPlay System implements the protocols specified in [MC-DPL4CS], [MC-DPL4R], [MC-DPL8CS], [MC-DPL8R], [MC-DPLHP], [MC-DPLNAT], [MC-DPLVP], and [MS-DPDX]. For further information, see Figures 9 and 10 in Section 6.3 Transport Requirements.

The specification defined in [MC-DPL4CS] describes the core protocol services of the DirectPlay 4 Protocol. The DirectPlay 4 Core and Services protocol facilitates communication between computer games for which a host computer manages the metadata of multiple computer game instances supporting multiple players. The protocol enables the implementation of functions to enumerate hosted game sessions and players, to add and remove game players, and to interchange data between game instances. The DirectPlay 4 Protocol also can optionally use the DirectPlay 4 Protocol: Reliable (as specified in [MC-DPL4R]) to manage network connections, to send and receive packets, and to perform reliable communication.

The specification defined in [MC-DPL4R] describes the functionality related to the reliable delivery of DirectPlay 4 messages. The DirectPlay 4 Reliable protocol guarantees message delivery and provides throttling for applications that use DirectPlay 4.

The specification defined in [MC-DPL8CS] describes the core protocol services of the DirectPlay 8 protocol. The DirectPlay 8 protocol provides functionality necessary for multiplayer game communication, including the ability to create and manage game sessions, enabling the enumeration of hosted game sessions and players, addition and removal of game players, and to interchange data between game instances over existing datagram protocols such User Datagram Protocol (UDP).

The specification defined in [MC-DPL8R] describes the DirectPlay 8 Reliable protocol and describes functionality related to the reliable delivery of DirectPlay 8 messages. The protocol is intended for

use in multiplayer game communication where it provides for the delivery of mixed messages, both reliable and unreliable, over existing datagram protocols such as UDP.

The specification defined in [MC-DPLHP] describes the technology available for enumerating DirectPlay 8 hosts and ports. The enumeration functionality provided by the DirectPlay 8 Protocol allows a DirectPlay Client/Peer to discover one or more DirectPlay Servers/Hosts.

The specification defined in [MC-DPLNAT] describes the technology available for the support of network environments that involve NAT. The NAT location functionality consists of extensions to the DirectPlay 8 Core and Service Providers Protocol [MC-DPL8CS] to improve NAT support.

The specification defined in [MC-DPLVP] describes the DirectPlay Voice Protocol. This protocol is used to provide optional voice communications for applications that use the DirectPlay System to communicate.

The specification defined in [MS-DPDX] describes the DirectPlay DXDiag Protocol. This protocol is used to provide connectivity tests and user diagnostics.

5.3.2 Member Protocol Groups

The calling game application and the DirectPlay 8 NAT Locator Protocol [MC-DPLNAT] provide the transport options for the system. The game application selects the appropriate transports and the necessary addresses to use. [MC-DPLNAT] provides the system with additional connection options for other DirectPlay Protocols which are available to [MC-DPL4CS] and [MC-DPL8CS] in order to establish connectivity.

In the DirectPlay 4 Protocol, the calling game application and [MS-NLMP] can be used together to provide a level of security to the DirectPlay 4 System.

5.4 System Internal Architecture

5.4.1 Communications within the System

Communications between the components of the system occur when protocol messages higher in the stack are treated as payloads of protocols below them. This hierarchy is illustrated in the diagrams in section 6.3, Figure 9 and Figure 10.

5.4.2 Communications with External Systems

If the application in a DirectPlay 4 System wants to use secure sessions, it is expected to communicate with an NT LAN Manager Authentication protocol-based system to establish accounts for authentication. The accounts are then referenced in messages described in [MC-DPL4CS] section 2.2.

Additionally, depending on the request of the calling game application, [MC-DPL4R], [MC-DPL8R], [MC-DPLNAT], and [MC-DPLHP] will communicate with the appropriate system transport. The options include IP, IPX, Serial, or Modem as illustrated in section 4.3.1.

The external systems are illustrated in section 6.3 where Application and Transport are the external systems to the DirectPlay protocols.

5.4.3 Incoming Interfaces

DirectPlay 4 and DirectPlay 8 Systems are designed to be generic game session management and transport systems for game applications. They are expected to provide the application with an interface similar to the interfaces documented in the DirectPlay 4 and DirectPlay 8 software

development kits for invoking the various protocol features and messages described in their associated protocol documents.

5.4.4 Outgoing Interfaces

For applications that use DirectPlay 4 System secure sessions, the [MS-NLMP] implementation is expected to provide authoritative, trusted responses to authentication requests made during the secure session join sequence.

Additionally all DirectPlay Systems need interfaces into the underlying selected transports (IP, IPX, Serial, Modem) in order to facilitate the transport of datagrams. This is illustrated in section 4.3.1.

5.5 Failure Scenarios

This section describes the common failure scenarios and specifies the system behavior in such conditions.

5.5.1 Connection Disconnected

The common failure scenario is an unexpected connection breakdown between the system and the external entities. A disconnection could be caused by the network not being available, or by one of the participants becoming unavailable. If the network does become unavailable for one of the DirectPlay systems in the session, the remaining participants remain active and expect the other party to continue with the communication pattern specified by the protocol. Similarly, in the case where one of the participants is not available, the remaining participants will continue to expect the communication to proceed as specified.

Generally, a protocol detects a connection breakdown failure through either of the following methods:

- By using a timer object that generates if the corresponding participant has not responded within a reasonable time span.
- By being notified by the underlying protocol that the connection is disconnected.

When a connection disconnected event is detected, it causes the protocol to tear down all related communications associated with that participant and follow the specification as defined in DirectPlay Core and Service Providers specifications for dropped players. The case of lost host connectivity triggers non-voluntary host migration as specified in [MC-DPL4CS] section 3.1.7.1, [MC-DPL8CS] section 1.3.6, and [MC-DPLVP] section 1.3.3.1.

6 System Details

This section contains the details that complete the descriptions in earlier sections of the document. These details are needed to understand and implement this system. Information already in the TDs should be referenced whenever available.

6.1 Architectural Details

This section provides two scenarios illustrating the use of the DirectPlay System. The scenarios are:

- Discovering, joining, and leaving a Local Area Network DirectPlay 8 game session
- Joining a DirectPlay 8 Host and Existing Peer With Network Address Translation Extensions

6.1.1 Discovering, Joining, and Leaving a Local Area Network DirectPlay 8 game session

This scenario provides a conceptual overview of the process of discovering an existing DirectPlay 8 session on a Local Area Network through the combined use of the [MC-DPLHP], [MC-DPL8R] and [MC-DPL8CS] protocols.

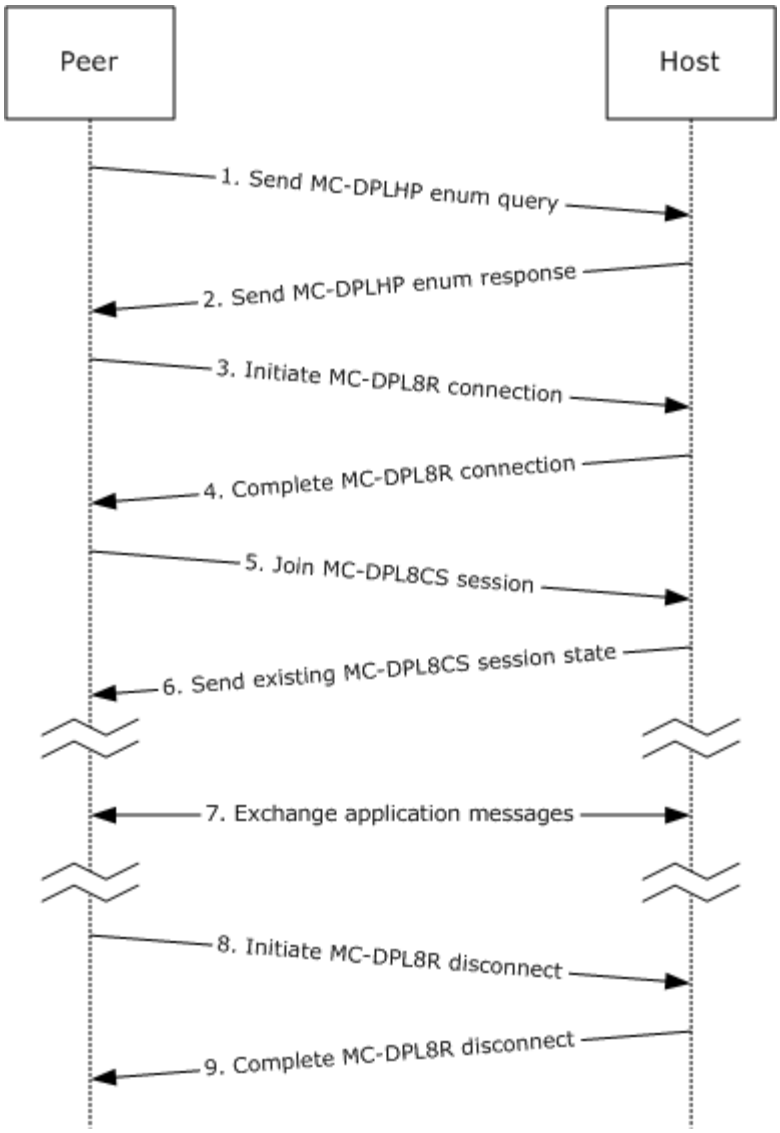

Figure 7: Example of the logical sequence of a participant's lifetime in a DirectPlay 8 game session

The logical exchanges that occur in this example are:

1. The peer sends an [MC-DPLHP] EnumQuery to the host directly or by broadcasting on the Local Area Network.

2. The host chooses to respond to the [MC-DPLHP] EnumQuery with an EnumResponse containing current information regarding the DirectPlay 8 session.

3. The peer processes the response and decides to connect to the host's session, and initiates an [MC-DPL8R] connection via a CONNECT packet.

4. The host accepts the [MC-DPL8R] connection attempt and sends a response. The peer will acknowledge that response and complete the 3-way handshake procedure. The messages exchanged are described in [MC-DPL8R].

5. Once the [MC-DPL8R] connection is established, the peer sends its player information to join the session as described in [MC-DPL8CS].

6. The host acknowledges the player information and sends the current game session state to the peer as described in [MC-DPL8CS].

7. The two participants then exchange application-defined messages using [MC-DPL8CS] and [MC-DPL8R] for as long as desired. If the connection is lost, the application determines the behavior on disconnection.

8. When the peer wishes to leave the session, it initiates an [MC-DPL8R] disconnect.

9. The host acknowledges the disconnecting participant and completes its own portion of the [MC-DPL8R] disconnect.

This returns both the host's and the peer's DirectPlay 8 system implementation to the states they were in prior to the peer joining.

6.1.2 Joining a DirectPlay 8 Host and Existing Peer With Network Address Translation Extensions

This scenario provides a conceptual overview of a nascent peer joining a previously established DirectPlay 8 game session with a host and existing peer, using the path test [MC-DPLNAT] extensions for improved firewall and Network Address Translation compatibility with the [MC-DPL8R] and [MC-DPL8CS] protocols.

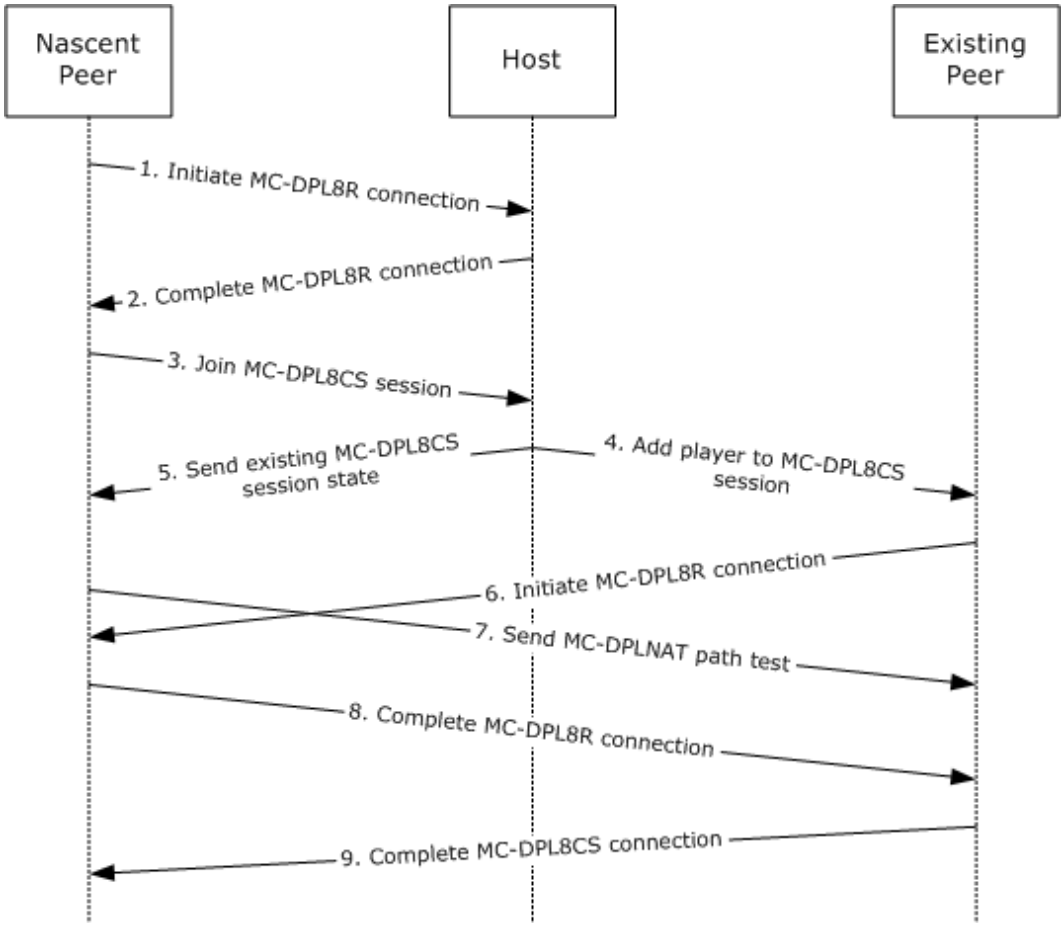

Figure 8: Example of the logical sequence of joining a DirectPlay 8 game session

The logical exchanges that occur in this scenario are:

1. The nascent peer initiates an [MC-DPL8R] connection to the host using a CONNECT packet.
2. The host accepts the [MC-DPL8R] connection attempt and sends a response. The nascent peer will acknowledge that response and complete the 3-way handshake procedure. The messages exchanged are described in [MC-DPL8R].
3. Once the [MC-DPL8R] connection is established, the nascent peer sends its player information to the host to join the session as described in [MC-DPL8CS].
4. The host reports this new [MC-DPL8CS] player to the existing peer and instructs it to connect.
5. The host also acknowledges the player information and sends the current game session state to the nascent peer as described in [MC-DPL8CS].
6. The existing peer attempts to establish an [MC-DPL8R] connection to the new peer using a CONNECT packet.
7. Simultaneously, the nascent peer uses the [MC-DPLNAT] extensions to issue a PATH_TEST to the existing peer in an attempt to establish any necessary firewall or Network Address Translation device mappings for the incoming [MC-DPL8R] connection.

8. When the nascent peer successfully receives the [MC-DPL8R] CONNECT packet from the existing peer, it accepts the connection attempt and sends a response. The existing peer will acknowledge that response and complete the 3-way handshake procedure. The messages exchanged are described in [MC-DPL8R].

9. Once the [MC-DPL8R] connection is established, the existing peer sends its [MC-DPL8CS] **player ID** to the nascent peer to complete the [MC-DPL8CS] connection and join attempt.

At the completion of this scenario, all three systems are participating in a game session and are able to communicate application data with each other using the [MC-DPL8CS] and [MC-DPL8R] protocols.

6.2 Communication Details

The DirectPlay System is capable of using multiple underlying transports, such as UDP/TCP, IPX, or serial connections. However, firewall and Network Address Translation issues only apply to UDP/TCP (IP-based) transports, therefore the [MC-DPLNAT] extensions are not relevant to other transports.

6.3 Transport Requirements

The following protocol stack diagrams indicate the relationship among the protocols within their respective systems. Protocols higher in the diagram use the protocols below them as their transports.

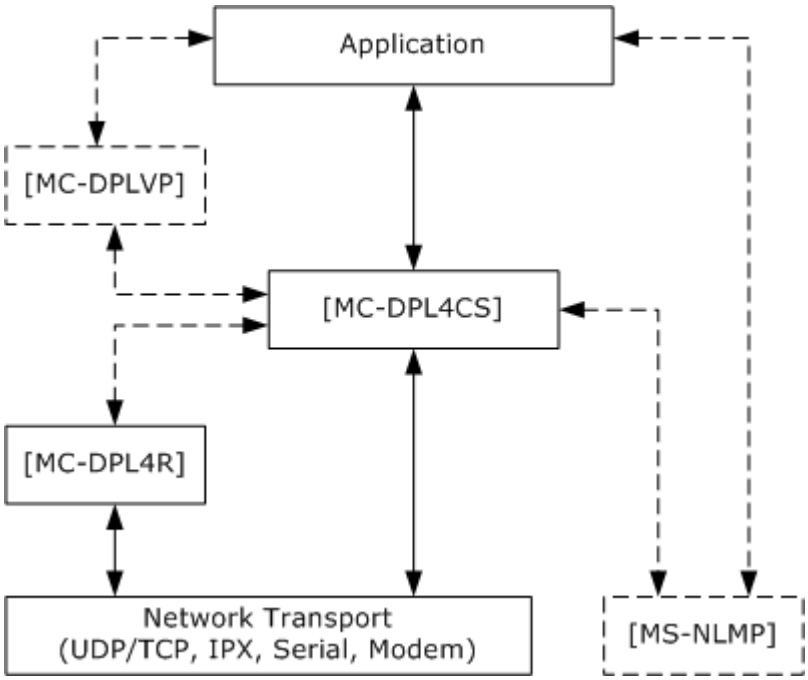

Figure 9: DirectPlay 4 System protocol hierarchy

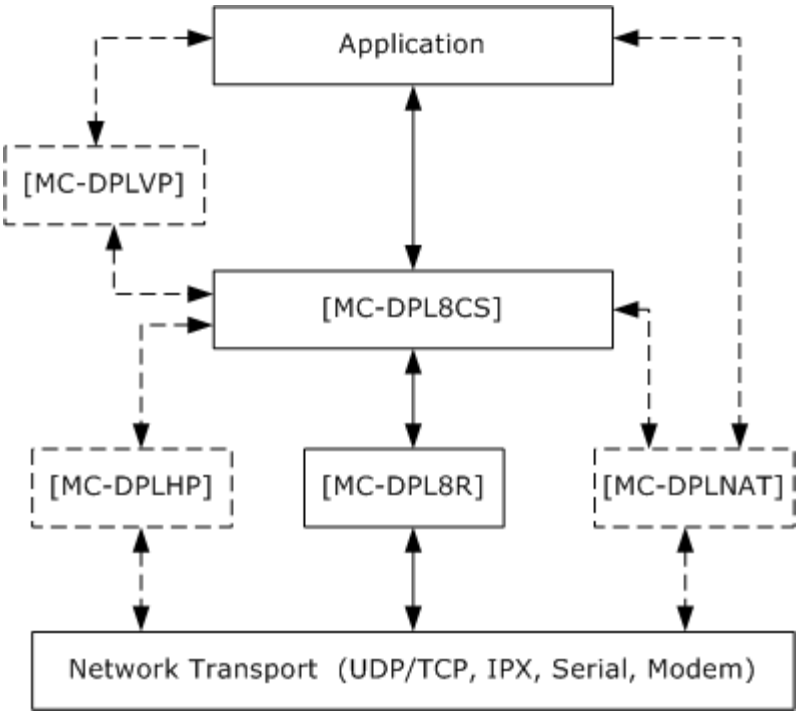

Figure 10: DirectPlay 8 System protocol hierarchy

In the DirectPlay 4 System, normal [MC-DPL4CS] operations and application-specified payloads are transported using [MC-DPL4CS] message types. If the application chooses to establish a game session with the reliable protocol option, the RP bit set in the Flags field of the DPSESSIONDESC2 structure as described in [MC-DPL4CS] section 2.2.5, and all application-specified payloads passed to the session management component will instead be transported using the reliable protocol described in [MC-DPL4R].

In the DirectPlay 8 System, [MC-DPL8CS] operations are always passed to the reliable protocol for transmission using the PACKET_COMMAND_USER_1 flag and not the PACKET_COMMAND_USER_2 flag, as described in [MC-DPL8R].

When the application has enabled voice functionality, it uses an API provided by the voice protocol implementation to invoke its functionality, as described in [MC-DPLVP]. The messages are then provided to the Core and Service Provider session management component for transmission. In the DirectPlay 8 System, [MC-DPLVP] operations are always passed through to the reliable protocol for transmission using the PACKET_COMMAND_USER_1 and PACKET_COMMAND_USER_2 flags, as described in [MC-DPL8R].

6.4 Timers

The DirectPlay System timers are confined to affecting only the associated Member Protocol in which they are documented. However, because of the largely vertical relationship among Member Protocols, a timer may cause a lower layer protocol to transition to a state that appears as an external non-timer event to a higher layer. For example, when the [MC-DPL8R] retry timer expires and determines that the maximum number of retries has been exhausted for a reliable message, as specified in [MC-DPL8R] section 3.1.2.5, the connection is considered lost and [MC-DPL8CS] must handle the associated disconnected player event through its documented procedures such as peer integrity checking or host migration.

6.5 Non-Timer Events

Because of the largely vertical nature of the DirectPlay System's relationships among Member Protocols, many external events apply equally to each protocol. For example, when the application layer wishes to join a DirectPlay 8 session, it initiates a connection in the [MC-DPL8CS] protocol, which in turn initiates a connection via the [MC-DPL8R] protocol. Once the [MC-DPL8R] connection succeeds, the [MC-DPL8CS] message exchange can proceed. Similarly, if an operating system environment change informs [MC-DPL8R] that communication is no longer possible, it should report it like any other connection loss to higher layers, which in turn must handle the connectivity failure according to their documented procedures.

Since each DirectPlay System Member Protocol encapsulates the state of those below it, the state of the system as a whole is effectively the same as the state of the highest level protocol.

6.6 Initialization and Reinitialization Procedures

Initialization and reinitialization of the DirectPlay System is controlled by the application. An application should be designed to use either DirectPlay 4 or DirectPlay 8, and only needs to initialize the corresponding Member Protocols: [MC-DPL4CS] and [MC-DPL4R] for DirectPlay 4, or [MC-DPL8CS], [MC-DPL8R], [MC-DPLHP], and [MC-DPLNAT] for DirectPlay 8. The optional voice transport, [MC-DPLVP] can be used with either DirectPlay 4 or DirectPlay 8. The protocols to be used should be initialized together, at the same time, since higher layers depend on lower ones and all share similar lifetimes.

Not all protocols have initialization procedures or are required to implement the DirectPlay System, however, and therefore may not need to be initialized at all. The Member Protocols without initialization procedures are [MC-DPL4CS], [MC-DPL8CS], and [MC-DPL8R]. The optional Member Protocols are [MC-DPL4R], [MC-DPLHP], [MC-DPLNAT], and [MC-DPLVP].

For DirectPlay 4, the order of initialization is therefore [MC-DPL4R], followed by [MC-DPLVP].

For DirectPlay 8, the order of initialization is therefore [MC-DPLNAT], followed by [MC-DPLHP] and then [MC-DPLVP].

6.7 Status and Error Returns

There are no specific error or status codes uniquely associated with the DirectPlay System failure scenarios.

7 Security

This section documents system-wide security issues that are not otherwise described in the Technical Documents (TDs) for the Member Protocols. It does not duplicate what is already in the Member Protocol TDs unless there is some unique aspect that applies to the system as a whole.

The DirectPlay 4 System supports limited NTLM authentication and encryption. This is described in [MC-DPL4CS].

The DirectPlay 8 System supports optional packet signing. This is described in [MC-DPL8R].

The DirectPlay systems are optimized for trusted peer-to-peer mode communication, and are best suited for applications primarily concerned with functionality and minimizing bandwidth usage. They are not recommended for uses where security is the primary concern.

The DirectPlay System allows an application to define simple passwords, typically supplied by the user through the application's user interface to avoid unauthorized connections to the system. The Client will need to provide the password to the Host before the Client is allowed to join the session. If the password is incorrect, the DirectPlay System rejects the connection attempt and returns an error. For more information see [MC-DPL8CS] and [MC-DPL4CS].

An application SHOULD always ensure that messages that it receives from the DirectPlay System are expected and are of expected length. Any malformed messages or messages from unknown Clients SHOULD be ignored.

The DirectPlay systems are not recommended for uses where security is the primary concern.

8 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft Windows® 2000 operating system
- Windows® XP operating system
- Windows Server® 2003 operating system
- Windows Server® 2003 R2 operating system
- Windows Vista® operating system
- Windows Server® 2008 operating system
- Windows® 7 operating system

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

<1> Section 2.1: The DirectX End User Runtime was incorporated into Windows 2000, Windows XP, and Windows Vista. It was made available to relieve users of the need to install DirectX End User Runtime when installing games.

<2> Section 2.1: DirectPlay 4 was released in 1996 in the DirectX 6 SDK and DirectPlay 8 was released in the DX8 SDK in 2001. There were some minor additions to DirectPlay 8 in the DX9 SDK but all the interfaces kept the same names. DirectPlay Voice was added in 2000 as part of DirectX 7.1 in Windows Millennium Edition

<3> Section 2.1: The DirectPlay System was officially deprecated by Microsoft in the DirectX 9 SDK in the summer of 2004. The libraries, headers, docs, samples continued to ship with the DirectX SDK until August 2007. Dynamic link libraries for DirectPlay 4 and DirectPlay 8 are present in the operating systems listed above.

Support for the DirectPlay Voice Protocol and DirectPlay 8 Protocol: NAT Locator Specification does not ship with Windows Vista, Windows Server 2008, Windows 7, or Windows Server 2008 R2.

9 Change Tracking Page

This section identifies changes that were made to the [MS-MGSO] protocol document between the May 2011 and June 2011 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the language and formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.

- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated.**

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1.2 References	Added explanatory statement regarding the removal of the publishing year from Microsoft Open Specification document references.	N	Content updated.

10 Index