SEECS

Distributed Computing

Protocol Design, Assignment #01

Ammarah Tauheed – 2010-NUST-SEECS-BE-SE-281
Faria Kalim - 2011-NUST-SEECS-BE-SE-259
Maryam Tauheed - 2011-NUST-SEECS-BE-SE-272

Type, Format and Sequence of Packets

Description of the Packet

Descriptor	Data types	Possible Values
IP of Sender	String(19 chars)	In the format "10.2.2.2"
Name	String(20 chars)	String without space
ID	Integer (32 bit)	6 digit random number
Sequence Number	Integer (32 bit)	Initially 6 digit random number
Acknowledgement Number	Integer (32 bit)	Initially 6 digit random number (equal to the sequence number of the first received packet)
Score	Signed 8 bit Integer	Minimum Score: -20 Maximum Score: 50
X position	Integer (5 bit)	0-31
Y position	Integer (4 bit)	0-15
Direction	Character (4 bit)	0-3
Status	Character (3 bit)	SYN, SYN/ACK,FIN, FIN/ACK, TAGGED, ACK ,TAG/ACK, R
Projectile X Position	Integer (5 bit)	0-31
Projectile Y Position	Integer (4 bit)	0-15

Handshake Sequence: New Player Joins the Game

1. Multicast Message (New player to all players)

Descriptor	Description
IP of Sender	IP assigned to sender
Name	Player's name
ID	Unique ID assigned to sender
Sequence Number	Random starting sequence number assigned to first packet
Acknowledgement Number	-1
Score	Current score (Can be initialized to zero, if player is just joining the game for the first time)
X position	X coordinate of player's position
Y position	Y coordinate of player's position
Direction	The direction that the player is facing
Status	SYN
Projectile X Position	0
Projectile Y Position	0

What if this packet is lost?

This makes no difference at the receiving end. However, at the sending end, a timer shall time out if no response is received after a certain time. Then the sender shall resend the packet. This shall continue until a packet is received or the timer times out three times. After three timeouts, the player shall wait a certain time before retrying the handshake.

What happens if the packet is received?

The receiving end adds a record of the new player to their game. It also maintains an "expected value" of the next sequence number in the next expected packet.

2. Reply (Old player to new player)

Descriptor	Description
IP of Sender	IP assigned to each receiver of multicast message
Name	Player’s name
ID	ID of receiver
Sequence Number	Random starting sequence number assigned to first packet
Acknowledgement Number	Sequence number of received multicast packet
Score	Current score of player
X position	X coordinate of player’s position
Y position	Y coordinate of player’s position
Direction	The direction that the player is facing
Status	SYN/ACK
Projectile X Position	0
Projectile Y Position	0

What if this packet is lost?

If the packet is lost, then the new player in the game does not receive a response. If there is no response from anyone, then timer at the new player’s end shall time out, and it shall resend the first multicast message. The receiving end shall receive a duplicate packet. That packet will be dropped as its expected value will not be equal to the current packet’s sequence number. Hence, it shall resend the last packet sent out to this player. (Please refer to the struct ‘Player’)

What if this packet is received?

- The receiving end sends out the confirmation packet.

3. Confirmation (New player to all old players)

Descriptor	Description
IP of Sender	IP assigned to each player
Name	Player’s name
ID	ID of new player
Sequence Number	Previous sequence number + 1
Acknowledgement Number	Sequence number of reply received from old player
Score	Current score of player
X position	X coordinate of player’s position
Y position	Y coordinate of player’s position
Direction	The direction that the player is facing
Status	ACK
Projectile X Position	0
Projectile Y Position	0

What if this packet is lost?

The old players do not receive an acknowledgement, and do not proceed to sending status messages. Instead they wait for the ACK. As the new player receives no status message, its timer times out, and it resends this packet.

What if this packet is received?

The players proceed to sending status messages.

Closing Sequence

1. Multicast Message

Descriptor	Description
IP of Sender	IP assigned to each player
Name	Player's name
ID	Unique ID assigned to each individual player
Sequence Number	Last sent packet's sequence number + 1
Acknowledgement Number	Sequence number of last packet received from recipient
Score	-1
X position	-1
Y position	-1
Direction	-1
Status	FIN/ACK
Projectile X Position	0
Projectile Y Position	0

What if this packet is lost?

The receiving players proceed as usual. However, when the sender does not receive a FIN packet, it resends another Multicast message when the timer times out.

What if this packet is received?

The recipient of the packet sends the FIN packet.

2. Reply of Recipient (the FIN packet)

Descriptor	Description
IP of Sender	IP assigned to each player
Name	Player's name
ID	Unique ID assigned to each individual player
Sequence Number	Last sent packet's sequence number + 1
Acknowledgement Number	Sequence number of recipient of the packet
Score	-1
X position	-1
Y position	-1
Direction	-1
Status	FIN
Projectile X Position	0
Projectile Y Position	0

What if this packet is lost?
Since the player who wants to quit the game does not receive a FIN packet, it resends its FIN/ACK packet.

What if this packet is received?
The game of the quitting player ends. All other player's remove the quitting player from their game.

Status Messages
These messages are sent out in two circumstances:

- The timer at the sender's end times out.
- The player receives another player's status message.

Hence, they also suffice as heartbeat messages.

Descriptor	Description
Sender's IP	IP assigned to each player
Name	Player's name
ID	Unique ID assigned to each individual player
Sequence Number	Last sent packet's sequence number + 1
Acknowledgement Number	Sequence number of recipient of the packet
Score	Player's score
X position	Player's x coordinate
Y position	Player's y coordinate
Direction	Player's direction
Status	ACK or R
Projectile X Position	X Position of the projectile that this player has fired
Projectile Y Position	Y Position of the projectile that this player has fired

A status message of ACK entails acknowledging the received packet, and providing new information about the player's rat's whereabouts.

A status message of R indicates that the rat has been tagged, and is going to be placed at a new random location. This message shall be sent for the random amount of time after which the rat is made to wait before it reappears in the game.

What if this packet is lost?
If this packet is not received at one end, the other player shall wait for a timeout and then send out their own status message. If three such timeouts are exceeded, the player shall assume that the other player has left the game.

In this way, if a player is left with no other players, then it shall repeat the handshake process.

What if this packet is received?
In response to a received status message, a player shall send out its own message to the sender.

Tagged! Message

Descriptor	Description
ID	IP assigned to each player
Name	Player's name
Sequence Number	Unique ID assigned to each individual player
Acknowledgement Number	Last sent packet's sequence number + 1
Score	Sequence number of recipient of the packet
X position	Player's score
Y position	Player's x coordinate
Direction	Player's y coordinate
Status	TAG
Projectile X Position	X Position of the projectile that this player has fired
Projectile Y Position	Y Position of the projectile that this player has fired

What if this packet is lost?

Until acknowledged TAG is received at the sender side, it should keep sending Tagged! packet. If there is no response after three timeouts, then we shall assume that the recipient is out of the game.

What if this packet is received?

The recipient shall send out an "Acknowledge Tagged" packet.

Acknowledge Tagged Message

Descriptor	Description
ID	IP assigned to each player
Name	Player's name
Sequence Number	Unique ID assigned to each individual player
Acknowledgement Number	Last sent packet's sequence number + 1
Score	Sequence number of recipient of the packet
X position	Player's score
Y position	Player's x coordinate
Direction	Player's y coordinate
Status	TAG/ACK
Projectile X Position	X Position of the projectile that this player has fired
Projectile Y Position	Y Position of the projectile that this player has fired

What if this packet is lost?

Until the tagged player does not receive this packet, it shall send out the tagged packet three times. If it does not receive this packet any of the three times, then it shall assume that the acknowledging party has left the game.

What if this packet is received?

The acknowledging party shall increment its score by 11.

The Structure "Player"

Data Members	Data Types	Description
Last packet received	*Packet	Used to store the player's information e.g. projectile direction, player's location etc. Also stores the sequence number of the player's last received packet. Hence we can find out the expected value of the next sequence number.
Last packet sent	*Packet	Resend to player if expected value is not equal to sequence number of received packet

Consistency

Handshakes

These are employed for a new rat joining a game.

Closing

This method shall be used if the user chooses to quit a game, or if they have lost (at -20 points).

Re-entering the Game after Poor Connectivity

If some packets are lost due to poor connectivity, we shall try to maintain consistency by allowing other players to wait for three time outs before removing the lost player from their game.

However, if a player loses all the other players in the game, then it shall re-employ the handshaking procedure.

Dropping Duplicates and Out of Order Packets

The sequence numbers shall be used to drop the packets that are out of order or are duplicates.

Assignment of IDs

IDs are assigned by using the rand() function, after seeding with time, to try and keep the IDs of all players unique.

One player per cell

If two players are directly in front of each other, then neither are allowed to move forward towards each other.

The Projectile

A rat can only shoot once in a certain time period. The projectile stops moving once it has reached either a rat or a wall. Two projectiles can share the same cell if needed.

Tagged!

The shooter of the tag is notified that a rat has been shot. This is determined at the client at which the rat was shot. Then the tagged player sends a Tagged! Packet to the shooter. If there are more than one shooters in a corridor, then the projectile that hits the tagged rat is the one that is closest to it.

Game Over

The game shall end for all players if one of the participants wins (has a score of 50).

Otherwise, the game for an individual shall end when it has a score of -20. In this case, the closing sequence shall ensue.