



SOCKET PROGRAMMING AND CONCURRENCY



Nathan Pascoe
256620

Socket Programming.....	3
Transmission Control Protocol (TCP)	3
Three-way handshake.....	4
Maximum transmission unit	5
Synchronous and Asynchronous Communications.....	5
Input and Output dispatch models	6
Interoperability	6
Socket Project	6
Requirements.....	7
Reference List.....	8

Socket Programming

The term socket programming happens to be the usage that belongs to an existing sockets that happen to be the mechanism behind the communication between different processes on top of the same machine or others that happen to be connected to an existing network. These sockets provide an existing programming interface that is going to belong to the network protocol on top of the operating system that has the ability exist as a used to initialise an existing socket as well as then send messages without the issue that belongs to packet framing or transmission control.^[1]

The OSI model happens to be used inside of handling the networking aspects that belongs to computing. This consists that belongs to 7 layers that happen to be application layer, presentation layer, session layer, transport layer, network layer, datalink layer as well as physical layer. The application layer consist that belongs to human-computer interaction where they access the network services. The presentation layer ensures that the data happens to be inside of usable format as well as happens to be encrypted. The session layer maintains the connection as well as happens to be responsible that is going to belong to controlling ports as well as sessions. The transport layer transmits data using transmission protocols such during the same time that TCP as well as UDP. The network layer decides which physical path the data is going to take. The datalink layer defines the format that belongs to the data inside of the network. Then finally, physical layer transmits raw bit stream over the physical medium.^[4]

Transmission Control Protocol (TCP)

There are several types of sockets such as stream sockets and datagram sockets that use distinct types of transport protocols for data transmission. The stream socket uses TCP and the datagram socket use user datagram protocols or UDP. The UDP is a protocol that does not have guarantees with the data transfer and can send messages without prior established connections are established. The TCP provides a more ordered and reliable connection in comparison to UDP between two hosts and that TCP guarantees that the message will arrive at the host in the correct order, this is achieved at the transport layer of the OSI networking model.^{[2][3]}

The TCP allows that is going to belong to transmission that belongs to information at the same time during the same time that inside of the computers has the ability send as well as receive data at the same time, this protocol uses packets during the same time that the units that belongs to data transmission which typically contain control information as well as consist that belongs to 1500 bytes inside of size. The network protocol previously stated control how the connection happens to be also terminated during the same time that well during the same time that the transferring that belongs to data. The control over this has the ability exist as a done by web browser as well as servers with each connection being identified by the endpoints that belongs to the client as well as server, with which the specific roles do not matter during the same time that they have unique identifiers that belongs to ip addresses as well as an existing port that is going to belong to each endpoint.^{[2][3]}

The TCP model works by establishing an existing connection between endpoints using the IP address during the same time that an existing unique identifier that being IPv4 or IPv6 as well as have the desire port enabled that is going to belong to the data transfer, this port allows that is going to belong to the operating system to assign connection to the specific client as well as server applications. The process inside of which this happens to be used happens to be the "three-way handshake".^{[2][3]}

Three-way handshake

The two-generals problem happens to be an existing problem regarding at the time where two or more computers happen to be inside of use as well as its states that two endpoints cannot simultaneously have common knowledge about the state that belongs to the connection. This happens to be meant to show that the TCP model cannot guarantee fully that the state between endpoints happens to be consistently connected as well as known. Then while this happens to be the case the three-way handshake aims to solve the issue with the exception of it being not able fully solve it only reduce the negative outcomes.^{[2][3]}

The “three-way handshake” consists of connecting using TCP, it firstly the client requests the server for a synchronise packet with a unique, random number. This number will ensure that the transmission can be fully sent in the correct order with no duplicates. Then if the server has received the segment, it will then agree to establish a connection by returning a synchronise-acknowledgment that would include the client’s sequence number plus 1 and transmit its own sequence number to the client. Finally, the client acknowledges the receipt of the synchronise-acknowledgment by sending its own acknowledgment packet, which contains the server’s sequence number plus 1. At the same time, the client can be transferring data to the server.^{[2][3]}

The termination process follows a similar step-by-step process begins with the client sending a FIN segment to notify the server that it no longer wants to send data and sends its own sequence number. Then the server acknowledges receipt with an acknowledgement segment that contains the sequence number plus 1 and when the server finishes the data transfer, it will also send a FIN packet with which it adds its own sequence number. Finally, the client sends an acknowledgement packet including the sequence number plus 1 which terminates the TCP connection to the server.^{[2][3]}

The TCP header contains all the data that is needed to complete the connection process and data transmission, the header data precedes the payload to be transferred and will typically consists of a data in the size of 20-bytes. Then followed up by a 40-byte data of additional information but this is optional and not used in packets. The TCP segments can send headers without payload data which are essentially pure headers and are also allowed if only acknowledgments, error messages and others that need to be transmitted in the case of synchronise and FIN messages.^{[2][3]}

The header consists of many components which are source port, destination port, sequence number, acknowledgment number, offset, reserved, flags, window size, checksum, urgent pointer and options. The source port which is 16 bits consists of identifiers of the port number of the sender. The destination port which is 16 bits contains the identifiers of the port number of receiver. The sequence number which is 32 bits specifies the first byte of attached payload data or is sent when the connection is established or terminated and for validating and sorting the segments after transmission. The acknowledgment number which is 32 bits contains the next sequence number that the sender is expecting with an acknowledgement flag used as a precondition for validity. The offset which is 4 bits specifies the length of the TCP header in 32-bit words to highlight the starting point of the payload data which can vary due to options. The reserved that is 6 bits is reserved for future use and that the field must always be set to 0.^{[2][3]}

The flags that are 6 bits contains the TCP actions for organising communication and data processing by using set states of flags to set or not set. These are used to set the states of urgent, acknowledgement, push, reset, synchronise and finish. The first state of urgent tells the TCP application that the payload data must be processed immediately up to the set urgent pointer. The acknowledgement is used in conjunction with the acknowledgement number for acknowledging the receipt of TCP packets. The push is used to ensure that the TCP segment is immediately pushed

through without being sent to the buffer of the sender and receiver. The reset is used when an error occurs and is used to reset the connection. The synchronise is used to represent the first step of the three-way handshake or connection. The final one, finish represents the signal to the other party to end the transmission.^{[2][3]}

The window size happens to be 16 bits as well as used to specify the number that belongs to bytes the sender happens to be willing to receive. The checksum happens to be 16 bits as well as happens to be so that transmission errors has the ability exist as a detected. The urgent pointer happens to be 16 bits as well as happens to be used to indicate the position that belongs to the first byte after the payload data that happens to be to exist as a processed urgently. The option has the ability contain 0 to 320 bits as well as happens to be used to include TCP functions that do not belong inside of the header such during the same time that maximum segment size.^{[2][3]}

Maximum transmission unit

The maximum transmission unit or MTU is the largest packet or frame size that can be sent in a packet or frame-based network such as the internet. The TCP uses MTU to determine the maximum size of each packet in any transmission and while using the ethernet protocol allowing 1500-byte packets to be the largest allowed in it while other network protocols have different MTUs such as token ring with 4464-bytes MTU. The issue that may be seen from this the sending of packets larger than the maximum size. This issue is minimized with fragmentation that breaks the packet into smaller fragments so that they can meet the maximum MTU which IPv4 and IPv6 allow for. The fragmentation allows for the fragments to transmit using the MTU parameter via the network layer in the OSI model and marks the fragments so that they can be unfragmented upon receipt.^{[7][8]}

The way it knows knowledge of this is stored in an IP header of a unit. The unit is equivalent to the MTU, for all intents and purposes it will be 1500-bytes for explanation. The unit consists of an ethernet header, IP header, TCP header, payload and FCS. The IP header, TCP header and payload can be identified as the packet and MTU and TCP header and payload can be identified as the segment while the payload is identified as the data and TCP maximum segment size or MSS.^{[7][8]}

The data, segment packet, frame and bit are the protocol data unit or PDU of different layers. Then in the same order session and presentation layer, transport layer, network layer, datalink layer and physical layer. The PDU can go from layer to layer and has different purpose on each layer. The bit are 1s or 0s and are used for the transmission of data. The frame the PDU comprises of original data with the TCP or UDP header, network layer header and data link layer header containing medium access control or MAC addresses of sender and receiver. The packet comprises of data with the TCP or UDP header and the network layer header containing logical addresses or IP addresses of sender and receiver. The segment contains data with the TCP or UDP header having the sender's and receiver's port numbers and is in the transport layer. The data in session and presentation layer and the PDU is purely data. Then finally in the application layer the PDU contains the original data made from the network application.^{[7][8]}

The process of reassembling the fragments must reassemble first before processing the PDUs that are sent along with the fragments. The IPv4 sender determine if they are fragmented or not in the bit with DF trigger of 0 and 1 in the IP header with the 1 being do not fragment. The IPv6 receivers do not fragment IPv6 packets instead relying on the source to fragment and reduce MTU size.^{[7][8]}

Synchronous and Asynchronous Communications

The synchronous communication happens in real time with two points communication information at the same time with each other. This requires the sender and receiver to be present at the same

time. The synchronous protocol where data is transmitted at regular intervals between the data bits. The types of applications that may use this can include but are not limited to chatrooms, video conferencing and telephone communications.^{[5][6]}

While asynchronous communication happens when information can be exchanged independent of time constraints and does not require the receiver to immediately respond and can respond at any time. The asynchronous protocol transmits one byte of information at a time at irregular intervals where it has a start bit precedes the byte of information and a stop bit follows the byte with a potential checking bit to be included in the transmission to ensure data integrity. The types of applications that may use this can include but are not limited to are emails, and forums.^{[5][6]}

Input and Output dispatch models

The synchronous and asynchronous input and output or I/O are threaded processes where with synchronous I/O it starts with the operation and immediately enters a wait state until the I/O requests have been completed and after waiting until it is completed, it will exit the wait and start the thread processing. The asynchronous I/O follows the same process except when it waits the asynchronous process will process other jobs while waiting and when it is completed it will interrupt those jobs and process the same way as synchronous.^[9]

Interoperability

The interoperability happens to be the ability that is going to belong to multiple networks, systems, devices, applications or components to communicate as well as this happens to be without the need that belongs to additional tools or interfaces such during the same time that TCP or IP usage inside of networks during the same time that it happens to be supported almost universally. This happens to be usually done by either the method where networks confirm to an existing protocol standard such during the same time that TCP or IP or by defining their own standard interface that is going to belong to every single one networks to adhere to or the other that belongs to providing an existing gateway to translate between the two protocols on the possibility that different happen to be used.^[10]

There happen to be steps that has the ability exist as a used to achieved interoperability at different stages that belongs to development. These steps happen to be that firstly, before any implementation barriers to interoperability have to exist as a minimized or even avoided. These barriers may exist as a proprietary standards as well as complex structures. Then it happens to be to concentrate upon the correct standardization approach, some examples that belongs to standards has the ability exist as HTML, XML and SQL among others as well as this step entails that the network ensure that the environment happens to be standardised across. The final step happens to be testing it, conformance as well as interoperability testing happens to be conducted inside of order to check equipment as well as network happens to be conforming to standards.^[10]

Socket Project

The design shall be created to be a chat application. This is going to use the additional services that belongs to socket.io to establish connections as well as node.js to establish the necessary server with the subsidiary node.js services that are express that is going to be used for framework as well as nodemon that is going to be used to complete automatic restarting to the node server at the time where changes happen to the code or file directory. The languages used would exist as a javascript, HTML, CSS as well as JSON. The coding language will follow the standard of camelCase where words are separated via the method of capitalization to indicate new words within the name.

Requirements

The features that belong to this service is going to consist of being able to join the server, send a message, leave the server, join as well as create rooms as well as lastly set usernames. The first feature would be joining the server which would entail being able to connect to a virtual server that is known as a localhost that is created by node inside of the terminal of that computer as well as it should load the initial HTML page. Then the ability to join or create a room except for before that they have the ability to create a username which would be up to user to input their own username with any characters that happen to be supported.

Then the next phase would be as previously stated joining as well as creating rooms, the joining part would consist of pressing upon hyperlink button to join as well as inputting a set that would contain the characters to identify the room such as "chatroom" or anything that the user has the ability think that belongs to. Then after joining the room the user would be able to send messages to anyone who happens to be currently connected as well as at the time anyone user who were leaving or connecting would have that information broadcasted to that room notifying inside of that room who happens to be connecting as well as leaving.

Reference List

1. Ibm.com. 2022. *IBM Docs*. [online] Available at: <<https://www.ibm.com/docs/en/i/7.1?topic=communications-socket-programming>> [Accessed 20 May 2022].
2. Research.ncl.ac.uk. 2022. [online] Available at: <<https://research.ncl.ac.uk/game/mastersdegree/workshops/networkintroduction/intro.pdf>> [Accessed 20 May 2022].
3. IONOS Digitalguide. 2022. *TCP (Transmission Control Protocol) – The transmission protocol explained*. [online] Available at: <<https://www.ionos.co.uk/digitalguide/server/know-how/introduction-to-tcp/>> [Accessed 20 May 2022].
4. Kumar S, Dalal S, Dixit V. The OSI model: Overview on the seven layers of computer networks. *International Journal of Computer Science and Information Technology Research*. 2014 Jul;2(3):461-6. [Accessed 20 May 2022].
5. Ibm.com. 2022. *Asynchronous Protocol*. [online] Available at: <<https://www.ibm.com/support/pages/asynchronous-protocol>> [Accessed 20 May 2022].
6. Ibm.com. 2022. *Synchronous Protocol*. [online] Available at: <<https://www.ibm.com/support/pages/node/758051>> [Accessed 20 May 2022].
7. Ibm.com. 2022. *IBM Docs*. [online] Available at: <<https://www.ibm.com/docs/en/ztpf/1.1.0.15?topic=support-protocol-data-units-pdus>> [Accessed 20 May 2022].
8. Ibm.com. 2022. *IBM Docs*. [online] Available at: <<https://www.ibm.com/docs/en/ztpf/1.1.0.15?topic=addresses-maximum-packet-size>> [Accessed 20 May 2022].
9. Docs.microsoft.com. 2022. *Synchronous and Asynchronous I/O - Win32 apps*. [online] Available at: <<https://docs.microsoft.com/en-us/windows/win32/fileio/synchronous-and-asynchronous-i-o>> [Accessed 20 May 2022].
10. Dl.acm.org. 2022. [online] Available at: <<https://dl.acm.org/doi/pdf/10.1145/234313.234424>> [Accessed 20 May 2022].

Bibliography

[video for socket programming](#)

