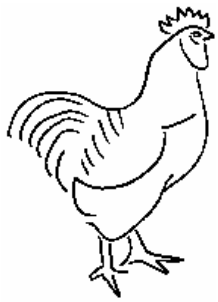
CLTC Project Information Sheet 4:

IRC − Internet Relay Chat



Developed by The 'Free Range' Community Linux Training Centre Project −
Version 1.0, August 2002. http://www.fraw.org.uk/cltc/

Most Internet services are one−to−one − but are disjointed in time between when messages are sent and received. Internet Relay Chat − IRC − is a 'real time' communications system between Internet users. For this reason it has special significance in the potential use of the Internet by society.

How IRC works

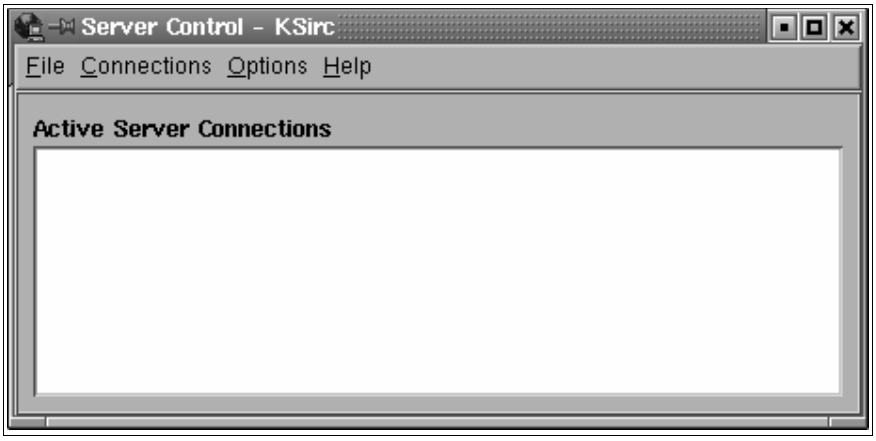
The very first communication between computers involved sending characters of text between two computers. As each character reached its destination, it was displayed on the screen. As well as being displayed, it could also be read and 'interpreted' by a computer − perhaps to trigger actions as a result of certain sequences of characters being sent.

In a more refined form this system was the basis of the early networks built around mainframe computers. Banks of 'dumb terminals' connected to the mainframe system. The text streaming in was displayed on the user's terminal. The characters produced by keys on the user's keyboard were sent back. Users could send requests to the mainframe, and the mainframe would send information back.

Internet Relay Chat − IRC − is a more complex form of this simple mode of communication. At the centre is an IRC server. Users connect to the IRC server, which opens a two−way communications channel − similar to the early dumb terminals. The users connect to the server with an IRC 'client' program − which organises information and maintains the connection to the server. Via the server, everyone is connected to everyone else − in real time.

All the IRC server does is receive text from the users, and then copy that stream of information to every other user. This is actually quite a problem for the server because it must maintain many connections simultaneously. With other Internet services, connections are only transitory.

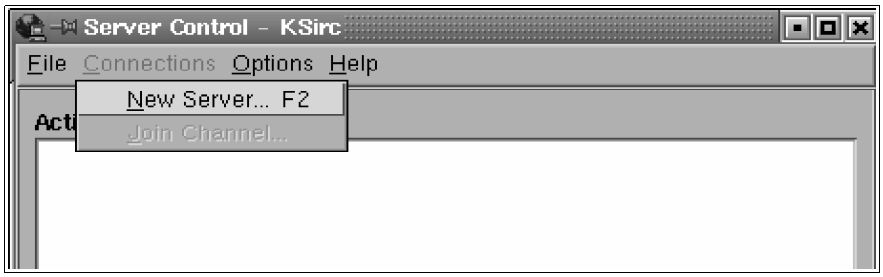
With IRC transmissions are arranged in 'paragraphs'. You can type text, and punctuate it, and can even sub−edit it. But it's not until you press the 'return' or 'enter' key that the information is actually sent. The server receives these paragraphs of information. It may receive a number at any instance. But they are 'relayed' back out to the users on a first−come−first−sent basis. This information then arrives at the users IRC client program and is displayed on the screen.



Using KSirc

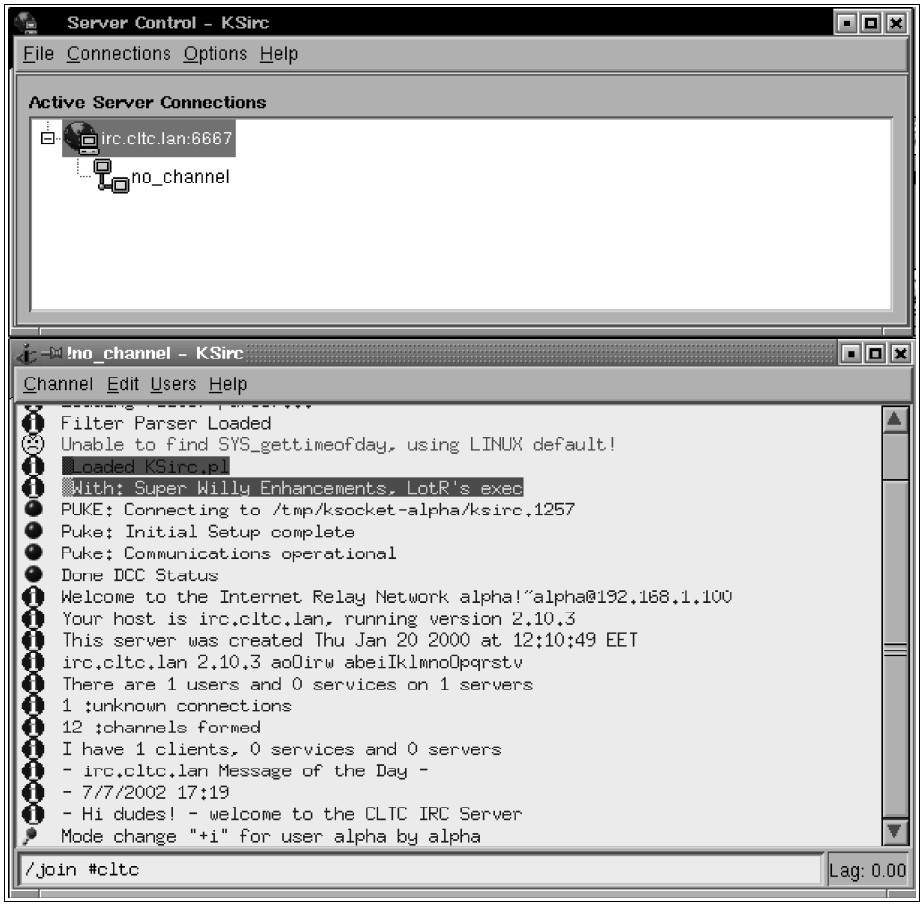
On the Community−Linux Training Centre system there is an IRC server. Each of the clients can practise using IRC − communicating with each other via the server. The client program used is *KSirc* (see above).

To start an IRC chat session, click on 'Connections' and 'New Server' − or just press F2. This opens up the 'Connect to server' dialog box (see next page).



The connection dialog requires that you give the address/name of a server – `irc.cltc.lan`. Leave the port number unchanged – unless otherwise specified, IRC servers always connect on port 6667. To initialise the connection just click 'Connect'.

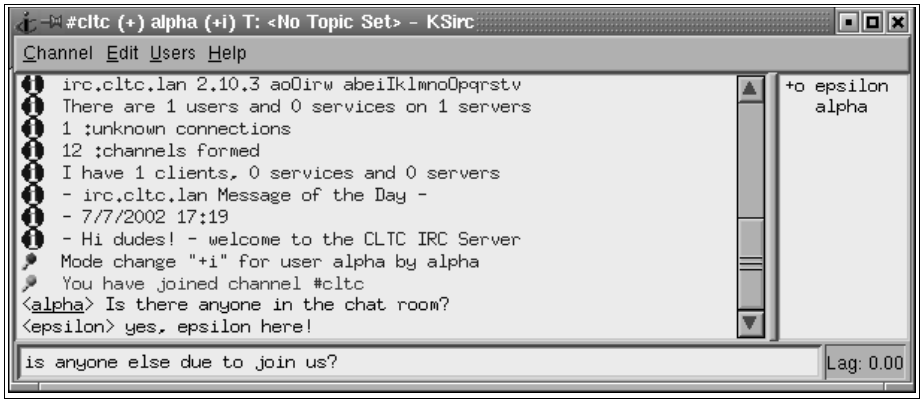
When you've connected to a server, the KSirc window displays a tree showing the name of the server, and a 'channel' name (see below). The reason that the KSirc window displays a 'tree' of connections is that you can open up multiple connection windows – connecting to different servers, and different channels on the same server, simultaneously. When the client connects, you also see a KSirc connection box. This will initially display connection information as the client program connects to the server.



Initially the connection window has the title 'no channel'. This is because you're only connected to the server, not to a chat channel. A single IRC server may operate many channels. How the server handles your request for access to channels varies from server–to–server. Some require that you are authenticated to use the server. Others are completely open.

The IRC server will respond to various commands. All the commands are prefixed with a forward slash – `/` – to differentiate them from ordinary text. You can get information on the commands available by entering '`/help`'. This will produce a list of commands. You then get help on specific commands by adding the name of the command to the help request. For example, '`/help ping`'.

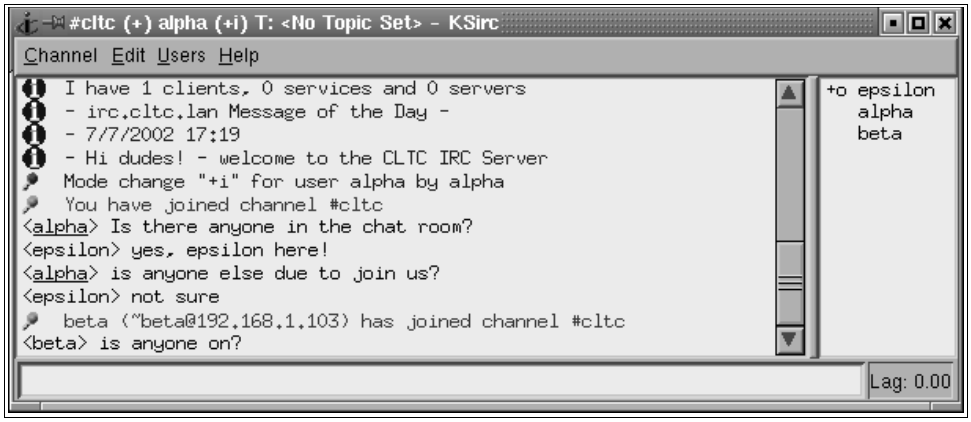
To connect to a channel you use the command 'join'. You then enter the channel name, with (usually) a '#' in front of it. The CLTC uses a channel called 'cltc' – so you enter '`/join #cltc`' to use it. You will then get a message telling you that you've joined, or that the connection failed. You will also notice that the KSirc window changes to show you are now connected to the 'cltc' channel.

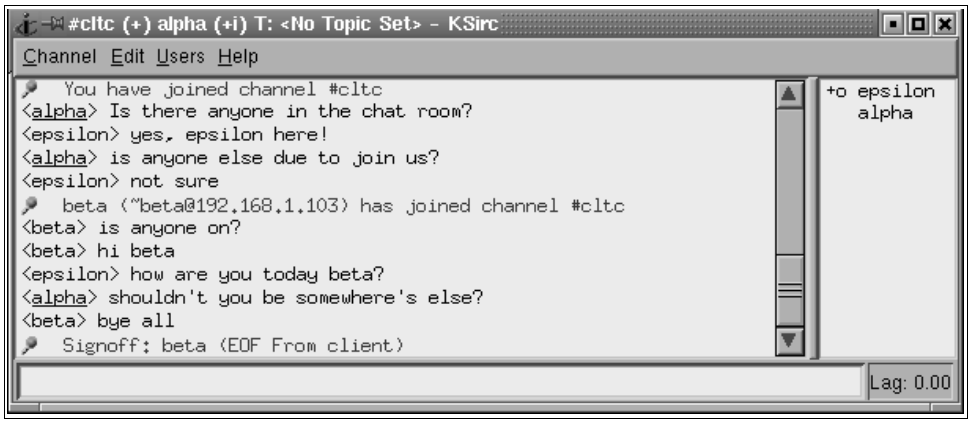


Once connected, you should also notice that the connection window has a new pane added to it, on the right. This lists the identities of the users who are connected to the channel. Most are just names. Some – in this case 'epsilon' – have special characters in front of the name indicating that they have some control over the channel (this gives them 'operator privileges' – e.g. kicking people off).

When you, or any other user, type in a message it is displayed on the client − prefixed with the identity of the user. If someone joins the channel, or someone leaves, the server will send a message itself to tell the users that someone has joined/left (see right).

To leave a channel all you do is enter '/bye'.





Using IRC for real

The CLTC is a means of practising the use of IRC, and understanding how it can help your work. IRC has many possible uses. From letting your family chat, to holding a conference between people across the globe.

To use IRC for real you'll need a server, and a channel. Some servers restrict connections, others don't. Some servers let you create channels (see below), others don't. The status of servers varies from month to month − but you could try `irc.afternet.org` or similar servers that allow users open access to channels.

When you connect to a server you can join a particular channel. But some servers allow you to create your own channel. The way this works is that if you '`/join`' a channel that doesn't exist, the server creates a channel with that name. In fact − this is how the CLTC system creates a channel for the clients to use. '*Epsilon*' − the name of the CLTC server− joins a channel on the server called '`#cltc`', causing it to be created.

When using the CLTCs IRC server you could try creating a channel of your own in this way, and then use it with the other users on the system.

'Java applets' and 'instant messaging'

IRC is a good system − but it has flaws. In particular the need to maintain lots of connections, meaning it requires a large amount of expensive bandwidth (connection capacity) at the server end to operate. It also has security problems. For this reason alternatives have been developed for the server operation, and for IRC clients.

A system you may regularly come across is the 'Java chat applet'. Java is a programming language. Applets are small Java programs that can be inserted into web pages. If you run the applet on your server, you can insert areas within a web page which work as IRC clients. This doesn't make your web server an IRC server. What it does is simplify connection to an IRC server on a system somewhere else by relaying that information as part of the content of a web page.

The only problem with the chat applet is 'Java'. Linux browsers don't support all Java functions. Therefore they have problems running the chat applet properly.

The other option in common use is 'instant messaging' − or 'IM'. This has been developed by some of the large Internet service providers as an 'add on' service for users. IM works differently to IRC because each user has a fixed identity on the server. For many this is one of the problems of IM because it is less anonymous than IRC. Users also require a proprietary IM client program to access the IM service. Users can then group themselves together to send messages − and talk together − as if they were on IRC.

The only other problem with IM is that you are often forced to take other information as part of the service − such as repetitive banner adds. It's also possible for those running the IM service to collect additional information about your usage of the system, and possibly use it for direct marketing. This is because the fixed identity you require to use the system gives a point of reference for matching the data collected about a persons use of the IM client. IRC also logs information, but this is usually the IP address and email address of each user logging onto the system.