UK Packet Conference held in Coventry, on 13th April, 2002

Present (as reported on the Attendance List):

G0BKN, G0EWH, G0KFS, G0MRH, G0CNG, G0NSW, G0SYR, G0TWN, G0WCI, G1AVF, G1BWT, G1CXE, G1DVU, G1HUL, G1IXV, G1LOA, G1ORG, G1SOG, G1YGY, G3MSW, G3OJI, G3XVV, G3ZFR, G4AFJ, G4APL, G4BBU, G4DIE, G4FPV, G4MTG, G4ROA, G4VYA, G6DZJ, G6KUI, G6TJZ, G6URP, G6VEY, G7BNK, G7CGB, G7JYF, G7RAZ, G7VBJ, G8ECJ, G8SFR, G8PZT, G8SEQ, G8TBF, G8YUP, M0DCM, M1EXO, M1FDE (50 participants)

Apologies were received (by G8PZT) from:

G0HDB, G1KQH, G1PLT, G3LDI, G3TIK, G4GUN, G4HIP, G4WYW, G6HJP, G7PUN, G7SRI, GM0HBI, GM1AHC, GM4JNB, GM4LNH, GM4PSX, GM4SUF, M1BFP, M1CUK.

Preliminary business:

The Conference was opened at 1030 by G1DVU, who welcomed participants, introduced himself and explained the arrangements for the day. Despite his initial reluctance, he was voted in as Chairman for the day. G7RAZ was elected Minutes Secretary.

G1DVU stressed the informality of the meeting, and the hope that each presentation would last 20 minutes with time afterwards for question and answer sessions. The agenda was to be essentially as per the one circulated by G8PZT prior to the conference.

Presentation: APRS, the Real Time Mode - Jim Andrews G1HUL

APRS (or "Automatic position reporting system") was in many ways completely different to standard packet operation. It was designed for real-time data flow only, with no store and forward arrangements, and needed no BBS stations. It was designed to be on one frequency, and provided its own infrastructure, with higher power stations helping lower power stations.

It was not just about GPS and positions. It was totally unconnected, existing within UI frames, and relying on redundancy of paths to get its beacon to the desired destination.

There were three components of APRS - Firstly the originating station, which sends out a beacon. Secondly, intermediate generic digipeaters, which could operate point-to-point, and thirdly the receiving stations.

Beacons could take many forms, but were designed to be a container for data (much like AX25). They had some pre-designed formats, but other formats allow application specific data to be sent transparently.

The main device required to run APRS was a device to generate the beacon. There were dedicated tracker / telemetry unit (eg Tiny Trakk or Mic-E / Pic-E), but also most recent TNCs would do the job. Even suitably equipped PC's, pocket, palm device or Linux systems would enable APRS packets to be sent and monitored.

An APRS beacon contained the sending callsign, the path, a symbol (if a positional beacon, e.g. car, truck, etc), and data. Receiving callsign was irrelevant.

Beacons could report a station's position (Lat/Long/Height), but also its speed, in the case of a car/truck, etc. This data could be input manually or obtained from GPS equipment.

Beacons could also be used for direction-finding. The station's kit output an APRS-compatible string, put a signal into a computer, TNC and radio - then triangulation was possible.

APRS could put "objects" on maps. These were "owned" by any station which cared to advertise them. They could mark events/places e.g. rallies, and could also "move" to be tracked by monitoring stations.

APRS could be used to broadcast weather information: wind speed, direction, etc. Also used for Telemetry, broadcasting anything you liked (alarms, environmental information, etc).

Although APRS was different to AX25, it was capable of sending messages - no large documents, but single line communication, generic bulletins, group bulletins. It had a system of acknowledgement packets as part of APRS protocol (but not AX25 RR/REJ frames).

The path for these messages need not be known. Instead APRS used generic paths, and when a station responded, the path was now known, so the next beacon used an explicit path. If that path failed, the system reverted to a generic path to find another route.

Queries and Status were APRS commands by which intelligent stations could be pinged and queried; some stations had databases of information which could be interrogated.

In order for APRS to cover a wide area “digipeating” was required. Local "relays" worked as Level2 hops, helping low power stations by filling in RF "black spots". Smart digipeaters had wider area coverage, with some being linked to the Internet (so-called "IGates") to give even further coverage.

Digipeaters had three generic aliases. "Relay" (the basic one), "Wide" and "Trace". Additionally APRS used "smart digipeating (using "WideN-N" and "TraceN-N").

With APRS there was a real need to know the path the packet had taken. If an explicit path could not be established, messages would continue to be sent as a general broadcast, producing considerable QRM. Ideally, when stations repeated, they changed the generic alias in the path to their own callsign.

G1HUL went on to illustrate how digipeating operated, showing examples of digipeated packets. WIDEn-n and TRACEn-n routines were shown, with successive stations transmitting diminishing Wide7-7, 7-6, (etc) packets.

All TNC's could do L2 digipeating, and alias substitution (except MFJ's). Later Kantronics, TNC-2 clones and Kenwood TM-D700's could perform smart digipeating.

At the receiving end, most computer platforms were catered for with suitable software. Additionally dedicated monitors were available (e.g. HamHud, D7 and D700). Typically these programmes displayed maps of an area, possibly linking into AutoRoute and other street atlases. These programmes showed output from weather stations, organized message management, had movement alarms, and could track stations.

The source of most confusion and positional was defective mapping. Generating your own maps required an understanding of how local mapping co-ordinate systems related to the world wide Latitude and Longitude system.

APRS used the GPS system reference WGS84 datum. But most maps were surveyed to their own country's datum (e.g. UK to OSGB36). Fortunately datum difference was at worst only +/- 250m, which was not normally critical. But most errors occurred due to poor map calibration. Ideally all stations would use the same datum when mapping.

On the RF side of APRS, the allocated frequencies were 144.800 FM (1200 baud) & 14.105 LSB (300 baud). Other frequencies existed, but unless you were setting up a closed system (e.g. for a specific event) there is no point in using any other than these two frequencies.

There was an increasing number of IGATEs. These stations took everything they heard on RF and put it onto the internet. Conversely, they listened to Internet APRS traffic and relayed anything which was unknown locally. These stations could monitor and relay several thousand station beacons within a few hours, and it was interesting to monitor such things as earthquakes (noting the reported Richter scale reading) and track paths of satellites.

On the net, station location queries could be made at http://www.findu.com.

APRS could be used for more than just hobby purposes. It could serve real time communication and carry out important data gathering. Monitoring could be conducted of significant weather "events", earthquakes, volcanic eruptions, etc. APRS could be put to emergency use (RAYNET, vehicle tracking, search and rescue, etc). It could help with pathless messaging, could enable remote status monitoring and could further propagation studies.

However it had a great appeal to many amateurs simply as part of their radio hobby. Many amateurs were using APRS (see the maps of observed stations). Perhaps packet decline was not as big as we thought - G1HUL showed how he had monitored over 570 stations from his QTH over a 2 week period.

For those with Internet the following URL's were recommended for further study:

http://www.aprsuk.net/

http://www.ui-view.com/

http://welcome.to/uiview/

http://www.ew.usna.edu/~bruninga/aprs.html

http://www.findu.com/

http://groups.yahoo.com (for ui-view, aprsuk and aprs groups)

http://www.tapr.org (for US-based aprssig and aprsnews groups)

During the question session, G1HUL agreed that the guaranteeing of message delivery was rather hit and miss over long distances with several hops. If such guarantees were required (e.g. for RAYNET use) a dedicated network of digipeaters was desirable.

Jim also agreed that most APRS use was confined to weekday evening and weekends, with relatively few users operating 24/7.

He tackled the issue of congestion caused by usage of one frequency, stating that congestion was not particularly an issue, although it would be if there more reliable links in the UK. Apparently US stations often discussed congestion.

Jim invited contacts for advice on APRS, advising use of e-mail in the first instance. His details:

Jim Andrews, G1HUL

E-mail: jim@stuckinthemud.org

Tel: 01530 249218

Presentation: XROUTER Network Infrastructure Software - Paula G8PZT

XROUTER, written by Paula, G8PZT, was node software, comparable in many ways to BPQ, but with superior performance and many more facilities. It may be of interest to ordinary packet users, to those who wished to establish a node for use by other packeteers, and to BBS sysops who wished to try out an alternative, more flexible RF front end to their systems.

More correctly termed a "packet routing system", XROUTER incorporated L2/L3/L4 capabilities (i.e. AX25, TCP/IP and NET/ROM), plus APRS functionality (with messaging shell), an inbuilt PMS, a chat server, and support for other packet applications such as WinPack, BBS programs, etc.

XROUTER was developed because of dissatisfaction with existing node software. Many nodes and the vast majority of packet mailboxes used G8BPQ's Packet Switch, but "BPQ" had major shortcomings, and its author appeared to have ceased supporting it:

BPQ suffered from: stability problems, crashing on frames > 256 bytes, running out of buffers, poor load balancing / session flow control. It had limited nodes table / circuits / routes / sessions, static timings, and data / nodes / MH corruption.

For the user, BPQ had no proper help, limited info, and no TCP/IP commands, Commands had to be in one packet, and the command line was not robust (it was easily crashed).

For node operators, BPQ was not friendly: you couldn't set the time / date, the configuration file had to be pre-compiled, there was no BUDLIST, and the VALIDCALLS size was limited. There was no watchdog, no screensaver, no remote control, a poor console interface, only a single console session, no chat server, no PMS, only single password, can't kill sessions, no domain resolution, no logging, no capture.

XROUTER was created to provide an improved alternative to BPQ, addressing each of the shortcomings listed above. The result was a piece of software which would fulfil the role of node (or router) and would offer many additional facilities to user and sysop alike.

XROUTER's hardware requirements were modest. Any PC from XT to Pentium, 640k memory, any video type (or run without VDU), an interface to the outside world (COM ports etc), and either a floppy, hard or RAM drive. It would run under DOS (3.3 onwards), DR-DOS, and Windows 95/98 (any version of Windows which had DOS emulation). It would also function with Desqview.

It would operate with the following hardware interfaces: COM ports (all types of UART chips), shared UART cards, 8530 SCC cards, Ethernet cards, keyboard & VDU, and any hardware for which an external driver exists.

XROUTER supported the following COM port options: KISS (all modes) SLIP, PPP, Baycom modems, YAM, Maxpack, NET/ROM backend, telephone modems, ASCII (dumb terminals) hardware / software flow control options. Its support for SCC cards included Baycom, DRSI, PA0HZP, Thor RLC100, PacComm PC100 / 120, and ITACARD.

Concerning AX25, it was compatible with versions 1&2 and also with Modulo-128 (EAX25). It was capable of digipeating, “digicasting”, and made intelligent adjustments of maxframe / paclen / frack, thus enhancing traffic throughput. Packet frames could be "piped" from port to port. Independent ID beacons could be set for each port, and per port BUDLIST and VALIDCALLS could be configured. There was port interlocking and split port operation. It offered comprehensive MHEARD lists, with adjustable sizes.

XROUTER had special NET/ROM features: Unlimited nodes / routes tables, dynamic L4 parameters, statistical multiplexing, proxy connections, nodes broadcast data validation, routing loop avoidance, echo, route record, INP3, and extended nodes / routes commands.

It also offered easily-configured TCP/IP features, as it incorporated an integral IP router, using datagram, and Virtual Circuit, NET/ROM, and encapsulation modes. Its TCP/IP commands were available to users, for telnet access and egress, with a DNS client and server, too. It was highly adaptable, supporting AXIP and AXUDP tunnelling protocols, dial-up networking, network address translation, IP masquerading, and Internet connection sharing, all respecting the RIP89 routing protocol.

APRS functionality

XROUTER's APRS features were considerable. It was capable of full "generic" digipeating, MIC-E encoding, and packet <> internet gating. It had a messaging shell for APRS and UI-VIEW. Its MH lists displayed APRS position, distance and bearing. It maintained best DX information and its ports could be configured to function with APRS only. It responded to both APRS and UIVIEW queries.

Its digipeater was capable of full generic digipeating - relay, wide, trace, widen-n, tracen-n - and had selectable digipeat modes, as well as pre-emptive digipeating. ID beacons could be digipeated, and even NET/ROM digipeating was possible.

Its messaging shell allowed non-APRS users to exchange messages with UIVIEW/APRS users, as well as reading bulletins and announcements. Message paths were selectable by the user, and duplicate prevention ensured appropriate use of bandwidth.

It also had an APRS IGATE. This allowed bi-directional packet <> internet gating, comprehensive traffic filtering, and independent filtering per direction. Unidirectional gating was selectable, and the software honoured NOGATE and RFONLY commands. It could connect to multiple servers, and offered automatic reconnection upon link failure, as well as blacklisting of dead servers.

Other additional features

XROUTER had an integral PMS, allowing the storage of messages to and from the sysop. This disk-based message store handled up to 65536 messages and, unlike TNC-based PMSs, allowed concurrent users. An additional PMS callsign and alias was possible. The PMS was accessed by PMS command, by L2 connection to PMS call/alias, or by L4 connection.

XROUTER also had a CHAT server. This could be accessed from the sysop's console or via AX25, NET/ROM, & TCPIP. It offered 256 local "rooms" (channels), 32512 public (server-interlinked) rooms, unlimited users per room, unlimited rooms per user, anarchic peer-peer linking topology, all with minimal setting up, plus activity logging.

XROUTER's command interface would be familiar to any BPQ node user, as the user commands were entirely BPQ-compatible. There was a comprehensive extended command set, with up to 16 command aliases, with built-in syntax help. It also offered hideable command aliases, a graduated help system, and an extendable INFO system.

Running an XROUTER node

An XROUTER node could be controlled by a sysop, both locally and remotely. Most parameters were hot-configurable. The system time / date was adjustable; there were multitasking DOS commands, and even a line editor for text files. XROUTER allowed simple programs and batch files to be run at the same time. Transmissions on any port could be disabled / enabled, and there were inbuilt manual pages, along with transaction logging.

Remote maintenance could be carried out via RS232, Ethernet, dial-up or radio links. XROUTER had a secure password system, offering individual sysop passwords. The node could be remotely restarted or rebooted. It even had an FTP server with secure login. For diagnosis purposes, trace display could be viewed remotely. External hardware control and monitoring was also possible (via parallel port).

The sysop was offered a console interface. It had a screen saver - automatic and manual, up to 5 independent consoles, fully configurable display options, a review buffer on each console, and an editable command line. Each console was ANSI colour compatible. There was also a DOS shell facility. There was even a console-less option (e.g. for remote nodes).

XROUTER enabled the sysop to trace packet traffic most comprehensively. All protocol layers were independently traceable. Ports, too, can be independently traced. Sent and received packets could be traced in different colours (the colours were configurable), and the whole traced activity could be captured to file.

There was a "Message of the day". As a precaution against system lock-ups, there were two "watchdogs" - both software and hardware - as well as comprehensive statistics to enable sysops to check the health of their system. All this is offered in addition to the proven reliability of XROUTER.

Sysops could run other packet applications with XROUTER. The application support required Windows or Desqview and a TSR, provided 8 and 16 bit BPQHOST API, and offered access via command line, AX25 and NET/ROM. It supported up to 8 applications, with up to 64 streams, and each application could have its own callsign, alias, name and L4QUAL. Access control was fully configurable, and applications could be in any order. There was also a comprehensive socket-style API.

Summary - miscellaneous strengths and limitations

XROUTER had been designed for maximum usability by BPQ-familiar operators. It used basic commands which followed the BPQ syntax, and its configuration keywords were similar to BPQ. It could use BPQKISS TNC firmware, and its nodes / routes save file used BPQ format.

XROUTER could be used in a wide range of applications. As well as being a QTH-based front end router, it could be a remotely sited router, offering support to BBS / APRS / Cluster / PMS usage. It could act as an APRS digipeater / IGATE, internet connection sharing router, or even as a Windows <> packet radio interface.

Its limitations were as follows. It was not a TSR (which would make it too big and complex). It required DOS or a DOS window. Its application support required Windows or Desqview. Linux & 32 bit versions were not (yet) available.

Furthermore, it only supported 25 lines * 80 cols * 16 colours. It didn't directly use expanded memory. Because it was a router, it has no SMTP, POP3, or BBS capability. It would not put a TNC into KISS mode. The source code was not available to the public. Its incompatibility with WW Convers server, however, was being reviewed.

XROUTER was obtainable by contacting the author (Paula) as follows:

Packet - G8PZT@GB7PZT.#24.GBR.EU

Alternatively, those with internet access could download XROUTER from:

Paula's website - http://www.g8pzt.pwp.blueyonder.co.uk

Support group - http://groups.yahoo.com/group/xrouter

Announcements regarding Clearance applications - Steve G8SFR

G8SFR advised Conference that, since Christmas, there had been an increase in the number of 70cms clearance applications being rejected. This was due to unspecified objections from the MOD, the primary user.

The frequencies mainly affected were at the top end of the 70cms band, although 433.650 had suffered from similar rejections. Even split frequency operation (where these frequencies would be used for receive only) were being affected.

G8SFR urged those who already had clearance to use these frequencies to do so, thus asserting our right to use what had been cleared. But he also advised users of frequencies in the 431 area that a possible shift towards 430 might become necessary.

Otherwise he advised that normal clearances were going through fairly quickly (10-12 weeks).

He also took the opportunity to reveal the new, simplified, application form available from the DCC for applying for Mailbox operation and clearance. He reminded Sysops of the need to have a current shutdown procedure in place, lest a check should be undertaken by the RA. Advice from the local RIS office about procedures was also advised.

Presentation - "AX25/IP integration experiences" - G3ZFR

Roger G3ZFR spoke of the desirability of increased cooperation between AX25 and TCP/IP packet operation - both at the level of frequency usage and also from a point of view of software applications.

He outlined the rise of both modes, but indicated how TCP/IP had declined more quickly over recent years, becoming perceived as a mode for enthusiasts only. He posed the question as to how TCP/IP could be made more popular for users, particularly given that most computer operators already operated many TCP/IP applications within Windows.

He asserted that, whilst not ideal on a user access frequency, AX25 and IP would co-exist on a point-to-point link. On shared channels however, due to AX25's aggression, TCP/IP would suffer under competition (and in the face of retries) but co-existence was feasible. Certain parameters may need tweaking, and the problem of gatewaying bulletins between the IP and AX25 BBS's still needed watching, but cooperation was the key.

He suggested that one solution would be for all nodes to support AX25, NETROM and provide IP routing. Local provision would offer users ideally at least one AX25 port and one IP port. An increase in high-speed links would be desirable, although he acknowledged the reduction in RF engineers involved in packet, and agreed that some very ambitious high speed projects currently being mentioned were beyond what was required - 9600 links would be quite fast enough to deal with current and projected levels of traffic. Mention was made of the high speed hardware to be exhibited later on in the day.

Presentation - "Enabling Windows users to access TCP/IP hubs over radio" - G3ZFR

Following on from his previous comments about everybody having TCP/IP courtesy of Windows applications (IE, Outlook, etc), G3ZFR went on to refer to his TNC2PPP EPROM.

This device, when inserted into a Tiny-2 (and other TNC2 clones), would allow it to be seen by Windows as a modem, and thus the native Windows IP application could be run - with very little modification - over radio.

Minor changes were still desirable. Users needed to adjust the Windows registry to ensure that broadcast packet frame length does not exceed 256 bytes. G8ZFR also anticipated further refinements of his own, particularly to avoid the initialising packets, put out at start up, which consume bandwidth unnecessarily. The use of SLIP as well as the currently supported PPP may be envisaged.

Users of Tiny-2 TNCs wishing to use Windows applications for IP over radio would need one 27C256 EPROM. The binary file for blowing onto it (along with instructions for installation and configuration) was obtainable, either from Roger (G3ZFR@GB7COV) or from his internet site (http://www.g3zfr.freeserve.co.uk).

G7RAZ also added that he had written notes for W98/NT/2K users who wished to set up SV2AGW's TCP/IP driver, enabling them to achieve them same as the above - i.e. to run Windows IP application (IE, Outlook, etc). Several participants availed themselves of these notes.

Presentation: PZT AX25/IP BBS Software - Paula G8PZT

Of potential interest to existing mailbox (GB7nnn) sysops, PZT BBS was a combined AX25 and TCP/IP mailbox "store and forward" system. It had all the features of standard AX25 BBS systems, such as FBB and NNA (including a file storage and retrieval system), but had additional TCP/IP functionality, similar to JNOS and certain Linux mail handling software - including a Web server, an FTP server, an SMTP server/client and a POP3 mail server. This meant it could serve both the AX25 and IP-using communities.

It could be accessed by any of the following interfaces; keyboard/VDU, serial ports, telephone modem, and Ethernet cards. Users and sysops could access it using a dumb terminal, a Hayes-compatible modem, a TNC, a serial link (using TCP/IP over slip), with Ethernet or dialup, via TCP/IP over radio, or via AX25 (including Netrom).

Hardware requirements were modest. On the hardware side, the minimum would be an AT compatible PC, with 640k RAM, a mono VDU, and a fairly fast drive (the BBS runs on a minimum of 3MB, with typically 40MB of working space being used). Additionally at least one COM or Ethernet port would be required to talk to the outside world - with radio or telephone interface hardware as required.

Software requirements were also simple. The operating system must be DOS3.1 up, or a DOS window in Windows. A RAM-disk was optional but useful. For AX25 use, BPQ or XROUTER node software was needed. If Ethernet was used, appropriate NIC drivers would be required.

PZTBBS had been designed with maximum compatibility in mind. It was fully compatible with other BBSs, was Y2K compliant, had a command set based on the W0RLI standard, used a standard BPQHOST interface, and worked with RFC822 mail compliance.

For the BBS users, it supported a range of common packet practices, such as mail collection via unproto broadcasts, as well as the /ACK method of obtaining acknowledgements. Users could use any major AX25 programme (such as Sally, Winpack, etc) and even Windows TCP/IP client programmes to access the BBS.

It has been designed with an emphasis on stability. By using DOS rather than Windows - as well as using the base 640k memory (to avoid memory contention) - it runs most reliably. Inside the programme were a host of error checking routines which ensure appropriate configuration, installation and use.

Speed had also been a design objective, and the programme used tight, optimised code. The use of indexes and a carefully designed directory structure minimise OS problems. By using the RAM-disk facility, even faster operation was obtainable.

A minimum of "Housekeeping" was also a feature of PZT BBS. No downtime was required for maintenance, since housekeeping was performed as a background task. No expiry routines were required, as the BBS used a fixed number of active messages, with a message number carousel, meaning that the oldest messages are eventually overwritten. The programme would also happily function on a fragmented hard disk.

Mail storage and retrieval

PZT BBS provided up to 20 message boards (mail areas), with single or multiple topics per area. Included in this were two areas for local, private mail and for mail in transit to other mailboxes. These areas are entirely configurable by the sysop (in name, contents and size). To enable access from automated packet programmes such as Winpack, there was also a single area compatibility mode.

Messages could be created in the standard ways, i.e. by direct entry (SP, SB, SR) and by creating messages from files. Optional signatures were available. Additionally, use can be made of a web form when using the HTTP interface.

Listing and searching for messages were done in the usual way. This could be forwards or backwards, as well as from and between certain points. The TO, AT, MID, and SUBJECT fields could be interrogated, and specific text could be sought. Exclusions could be incorporated, lists could be multi-parameter, as well as "all" or "recent". Again this could be done additionally via a browser interface.

Messages could be retrieved by being read at the command line, by using the HTTP interface, or by using POP3. When reading them, this could be with or without full headers, and required the number of the message only. PZT BBS allowed multiple reading, the reading of new mail only, as well as HTML to text translation, and pagination.

Messages were forwarded between BBSs using a variety of formats. PZT BBS was capable of MBL forward-reverse and reverse-reverse, as well as FBB block and compressed forwarding. The programme had an SMTP server and client, which incorporated bi-directional SMTP <>NTS gating facility. Mail could also be exchanged by telephone, or telnet. Mail prioritisation (smallest first, SP before SB, etc) was incorporated. Additionally, mail can be forwarded to file, for emailing or posting.

Network problems were fully catered for. In the event of a normal forwarding route failing, PZT BBS had a facility for route fallback. Repeatedly failing peers were blacklisted, with the conditions for the blacklisting being configurable. Sysops were also notified of ping-pong messages, unknown routes, and items of looped or rejected mail.

Various additional sysop controls of mail forwarding were: forward upon receipt, control of day, time, size, type, port, as well as separate BBS & PMS forwarding runs, configurable run intervals, transaction logging, R-line stripping, and 3rd-party to private mail conversion.

PZT BBS was also capable of generating targeted bulletins (for example SB RALLY @ 44.GBR or SB ALL @ JPN.AS), which would be tunnelled to their destination area, instead of using "flood" distribution. This method left no "snail trail" on PZT systems - the in-transit bulletins being invisible - and required no special setup. If adopted by other authors, it would make FBB's REDIST server redundant.

File storage and retrieval

Just as with messages, so also files could be stored and then accessed by various methods. From its W, U and D commands, PZT BBS switched into “files” mode - using file "areas", each of which could have sub-areas. The BBS had an integral filing system (PZTDOS) with DOS-like commands. The file transfer protocols supported were: ASCII, YAPP, XModem, Xmodem-1k, as well as real-time 7plus.

In line with its IP compatibility, PZT BBS allowed FTP up/download. The FTP server permitted both passworded and anonymous access, and the sysop could configure each user’s access rights individually (e.g. root directory rights, read/write/execute rights). User groups could be set up, and the FTP server accepted “/” and “\” in any combination. It had an extensive command set (35 commands), and was compatible with a large range of Windows FTP clients.

Additionally the HTTP interface could be used for file retrieval.

Services and servers

PZT BBS had a cross between the WWW and TeleText, called "TextWeb". This facility offered users variable sized pages, numbered 1-999, which incorporated plain text, ANSI or HTML. It offered freedom of layout, and users could read existing pages or upload new/updated pages of their own. It had an automatic page updating facility.

PZT BBS has an HTTP server, which was compatible with IE5, offering static and dynamic pages. It incorporated Perl-like script commands for the BBS interface, as well as transaction logging.

The programme had a comprehensive White Pages database, which included users' email addresses. It featured extensive data validation algorithms, with automatic refresh and update to next level, lockable records, a record editor, and comprehensive search facilities.

PZT BBS had several inbuilt servers, such as: ECHO, PING, REQBUL, REQDIR, REQFIL, REQCFG and WP. Additionally it worked with various external servers, such as LSTBUL and LSTSRV, with some 3rd party servers (e.g. AUTO7P) also being catered for.

The programme also featured "Doors". These were external programs designed to encourage user interactivity. Examples might be: an e-mail program, an adventure game, NGR <> QRA converter. Anybody could write suitable "doors".

There was also a Gateway facility, by which users can connect from port to port, or from mode to mode. (E.g. a radio user could connect to a telephone port, or a TCP/IP user could switch to a Netrom port). As in other areas, access to the gateway was controlled by the user privileges, which in turn are configured by the sysop.

Features for sysop usage

Sysops could offer users 4 prompt levels: novice, verbose, regular, and expert, each offering different levels of support. Users could choose to receive data which included ANSI colour. There was a comprehensive help system. For security, there were optional passwords, and for ease of reading, there was optional pagination, with selectable lines per page. Users could be offered callsign aliasing, to enable those with two callsigns to collect both sets of mail at one session. There was also a Ping and Telnet shell.

Sysops had at their disposal a comprehensive set of DOS-type commands, for file management. They could review held mail, and when doing so, are presented with the reason(s) why the message is being held. They could control forwarding runs, and kick off users.

For help and diagnostic purposes, there was an inbuilt sysop manual, a forwarding error log, facilities for session tracing or capturing, and (for testing) sysops could simulate a user login. They could import and export mail, open and close ports, and rescan mail. Depending on the level of changes made, they could re-initialise or restart the programme. They could even reboot the computer remotely, if desired.

Full editing facilities were available with PZT BBS. The sysop could make use of an inbuilt line editor for any text file. User records could be created / deleted / edited. There was also a message header editor, a WP record editor, and a password privilege editor. All aspects and fields of messages could be edited - type, status, to, from, at, subject, and message text.

PZT BBS had integral message filtering, which examined both subject and content. Depending on its configuration, it would hold locally entered mail, optionally allowing validated users to bypass local hold. Alternatively the sysop could configure the filter to auto-hold messages, depending on any of the fields (type, to, from, at). Holding could also be done manually. Held messages were flagged on the BBS status line and at the sysop prompt, inviting him to review, release or delete as appropriate. PZT BBS would also hold on R-lines - thus allowing the filtering of internet imports, out of date messages, etc.

Messages could be exported for file either automatically or manually. Each message could be exported to many files, in overwrite or append modes, along with exporting in MBL or plain text mode. Exported mail could contain full, first or no R: lines, with wildcard matching on: type, to, from, at, MID, subject or content.

Messages could be imported from file, either at start-up or on request, or every so many minutes. Messages were accepted in MBL format (compatible with other BBS), and importing was a background process, enabling the BBS to continue running as normal.

PZT BBS had a versatile file management capability. Sysops could copy messages to file and make messages from files. With PZTDOS, all files could be listed, copied, created, moved, renamed, deleted, and viewed. Directories, too, could be created and deleted. For remote sysoping, there was a line editor, and at the BBS console, a full screen editor was available.

The BBS Console message editor was full-screen and fully multi-tasking (ie. operating entirely concurrently with the functioning BBS). It edited a copy of the file (for safety), using a Wordstar command sub-set, and had extensive cursor movement controls, allowing the sysop to block copy, move, delete, write to file, insert from file, and search.

The Console also enabled the sysop to shell out to DOS from there, have terminal sessions, and chat with users. A bell sound was configurable to allow the console to relay sysop paging. Similarly it had an optional connect / disconnect bell. A simple message browser was also available.

The Console's main function, however, was to display BBS status information, which it dis via a split screen layout. Every port had a window, for monitoring the activity of each connected user. Each window had configurable colours and positions, and there was an additional scroll-back facility on the sysop's window.

For remote operation, PZT BBS provided a "virtual console", offering an exact image of the main console, along with the same facilities. This could be enabled and disabled at will over an RS232, dialup or Ethernet connection, and required only an ANSI terminal.

PZT BBS enabled the sysop to feel in control of the mailbox. It was extensively customizable, with configurable texts, colours, information files, etc. It also offered highly configurable security levels and user privileges, with separate passwords for radio, telephone, POP3 and FTP access.

Benefits and limitations

The principal advantages of PZT BBS were: the multiple mail areas, which facilitated the selection of desired reading material; the integration of AX25, IP and PSTN, which encouraged maximum participation in a BBS, and encouraged network cooperation; the remote control and virtual console, which allow considerable ease of sysoping; the IP facilities (web browser interface, SMTP, POP3, FTP), which satisfied the IP enthusiasts as well as catering for users of native Windows applications; the stability and modest requirements of the programme, which reduced key burdens of sysoping; and finally its rapid development undertaken by a responsive local author, which generated confidence amongst its user base.

On the side of perceived disadvantages, WA8DED host mode was not yet supported, it only ran under DOS (or in a DOS window). Linux and 32 bit Windows versions not expected in the short term. There was no support for NNTP (although this could be addressed), and there was no support for PPP linking (to come). To quote the author, "It's a BBS - not a TCP/IP "hub".

PZT BBS was obtainable by contacting the author (Paula) as follows:

Packet - G8PZT@GB7PZT.#24.GBR.EU

Alternatively, those with internet access could download the programme from:

Paula's website - http://www.g8pzt.pwp.blueyonder.co.uk

Support group - http://groups.yahoo.com/group/pztbbs

Presentation - "Thames Valley IP Group Projects" - G8ECJ

Robin, G8ECJ introduced the group, their preference for IP with Linux, and outlined a recently developed piece of hardware, the G4XYW 9k6 modem, based on a Atmel AVR RISC processor, and intended for the OptoSCC board. A sample was circulated.

The modem would be of particular interest to those needing to link to stations equipped with Kantronics 9k6 TNCs in the light of the work done on the transmitted wave form. It was claimed to be marginally better than an RUH 9k6 modem.

To improve performance they may introduce a couple of extra chips to improve the interfacing of the modem to the PC's RS232 port. They will also be developing an interface to USB ports, given that the RS232 standard may not be around for very much longer.

Secondly he described the group's Regen controller, based on modem hardware. It keyed up on squelch open or DCD from modem selectable active high or low. It sent flags when no data or when handling invalid data, detecting Morse idents etc and masking them out. It did CWID, which was programmable on a per CPU basis, although its transmission was time dependent and could crash a packet frame already in transmission.

Thirdly G8ECJ demonstrated a Front end processor board, suitable for installing inside a PC. It contained a 64180 chip as used in some TNCs, 2 SCC chips, giving scope for 4 radio ports. It could be connected to a modem (Baycom or G4XYW) or via SLIP to a Windows PC, or via KISS to an existing TNC. A few were offered for sale at low cost...

Fourthly a PC motherboard was exhibited, which was a self-contained 4 port node with processor on board. These were being offered free of charge.

Finally G8ECJ described the "Morelia" BBS software, written in Python (for Linux users) by Andy G4XYW. The BBS software - still in beta stage - aimed to interface both AX25 and IP users to news and mail. More details would become available from the TVIPUG in due course.

Presentation - "The MAXPAK local area network" - G0CNG

Chris, G0CNG, chairman of MaxPak and sysop of GB7MAX, outlined the main nodes and BBS's in the Midlands. Centred on GB7WV in Wolverhampton, he explained the role of the network as it linked to GB7MAX and the BBS's radiating out from it. He bore with good grace the couple of observations made about the use of some frequencies and callsigns not in line with DCC guidelines.

Presentation - "The MAXPAK MAX02 packet modem" - G4VYA

Joe, G4VYA explained the origins of the modem. It was his design, adapted from the YAM modem designed by Nico Palermo. It could use either the power supply from the PC or from an external source. He recommended the use of AGW software, as this enabled the tnc to be configured to be a 1k2 or a 9k6, via the program's internal tnc commands - ie. no hardware adjustments were required when changing from a low to a high speed modem.

He also recommended visiting the Maxpak web site to download a slightly doctored version of AGW, which would overcome some problems experienced by users of Windows 95.

Discussions

The Conference had run significantly over time. Participants were beginning to need to get away as the discussions were beginning (at 1700).

G1DVU attempted to chair the consideration of the first topic ("Packet Radio, present and future") and some brief general observations were made. However, with initially discreet departures becoming less discreet and with informal discussion groups which were becoming more vocal, G1DVU wound up the proceedings, thanking among others Roger G3ZFR for organising the venue, and especially Paula G8PZT for having proposed the event and having cajoled people into coming and participating.

Minutes by Mike Wager, G7RAZ @ GB7WIS.#24.GBR.EU

MikeWager@aol.com