MANAGED FILE DISTRIBUTION ON THE UNIVERSE NETWORK

Christopher S Cooper

Rutherford Appleton Laboratory, England

"Where there is no doubt, deliberation is not excluded as impertinent unto the thing, but as needless in regard of the agent, which seeth already what to resolve upon."

Richard Hooker (1554?-1600)

Abstract

The file distribution system on the Universe Network consists of a distributed set of co-operating agents which provide clients with a reliable bulk file collection, transfer and delivery service. The agent systems incorporate specialised techniques for optimising use of the satellite channel, as well as making available facilities for broadcast file distribution. The distributed system architecture and protocols are described, with emphasis on the separation of control and data transfer. A detailed presentation is given of the way in which agents interact with both hosts and other agents to achieve both reliability and robustness in the face of breaks in host or network availability. A description is given of the special transfer methods used and some of the possible applications for such a system. An indication is given of possible future developments to enable evolution from an experimental to service system.

1. Introduction

A topic of fundamental importance in computer communication is the reliable transfer of bulk data, most often in the form of files, between one computer and another. Of increasing importance is the ability to transfer digital data of growing diversity: examples range through voicegrams, bit mapped images (including facsimile), graphic displays, teletex and videotex, as well as more conventional textual material and programs. The advent of local area network technology offers the potential to distribute all these forms of information in a uniform way. The office exemplifies an environment where many of these ways of accessing and displaying information have application. Typically, in an environment of this sort, the details of using an appropriate communication mechanism are of no interest, and may indeed be a hindrance to an end user. In practice, the exigencies of a particular protocol mechanism may be only one aspect against which insulation is required. In the case of one office system transferring a document to another, it may be very important to insulate the user as much as possible from temporary severance of parts of the network and from unavailability of the remote station. The cause of the latter may range from component failure, through dedicated use of personal computers, to time zone difference and different office hours. In all these cases, desirable features of a workable system include being able to delegate the task of transfer with some confidence that, even if it cannot be accomplished immediately, responsibility will be taken and the user may proceed in many respects as if it were accomplished.

1.1 Environment for file distribution system

The Universe network [Leslie84], [Leslie83], [Kirstein82], [Adams82] is one in which local area networks at various sites are linked by broadcast satellite to achieve a wide area high speed network with low error rate. Construction of the network began in 1981 and by mid 1983 there were some 150 computers attached to the network, ranging from a large IBM 3081D mainframe, through a number of GEC 4000s, PDP11s, some Primes and a VAX, to a large number of micro computer systems, based on 68000, 8086, Z80 and 6502 cpus. In the

early stages of the project, these machines were used to perform a variety of bilateral experiments, many associated with establishing the network, its basic network protocols and providing an infrastructure in which more complex distributed systems experiments could be undertaken. The prototype system described here is the outcome of one of these "second generation" systems experiments.

The linking of two constituent networks in Universe is performed by a bridge [Leslie84], [Leslie83], which forwards traffic between the networks on a per block basis. The point to point network services offered by Universe are of three classes.

(i) Single Shot, in which a single request block (packet) is transmitted across the network to a named service and a single reply block is expected in response.

(ii) Open Connection, in which a full duplex route is established to a named service by an initial exchange similar to Single Shot. Blocks sent on this route are guaranteed not to be reordered; there is a small probability that some may be lost. The probability that a received block will be corrupt is negligeable. There is neither flow control nor recovery of lost blocks provided by the network on such a route. Congestion relief in the bridges is achieved by discarding blocks and is the most likely cause of lost blocks. This trade off in the design was taken in the interests of achieving high speed, particularly in the bridges.

(iii) Datagram, in which a single block, containing both data and Universe global destination address, is conveyed by the network, with no state information retained by the bridges.

Another class of network service on Universe is broadcast [Waters84a]. The facility is available to clients on a series of independent channels allocated by the service. A given channel is accessed by Open Connections from the originator(s) of the broadcast data and from those clients wishing to receive data on that channel.

1.2 Motivation for file distribution system

A prime objective of the Universe project is the exploration of the application and extension of distributed computing techniques currently in use on local area networks to a wider area. A strong motivation for the development of the current system was the desire both to test such techniques and to provide a suitable test vehicle for the development of bulk transfer mechanisms appropriate to the satellite segment. Several important properties of the Universe network stem from the use of broadcast satellite techniques to achieve wide area coverage. Of particular relevance in the current context are

(i) the direct use of broadcast to enable simultaneous transmission of data to several sites;

(ii) the cost of the satellite segment, making it desirable to optimise its use;

(iii) the incidental property of the network being severed more often than not because, being an experiment, the satellite is available on a very limited schedule.

While property (iii) might not be present intentionally on a service network, its presence in the Universe network has been turned to advantage by ensuring the development of a system which can survive such interruption and providing a testing environment. While optimising use of satellite bandwidth and using the broadcast facility are each important in providing a useful and economic network service, they are both sufficiently technically specialised that an end user system would expect to be insulated from the details.

2. Agent philosophy

An exceedingly general way of achieving an objective is to subcontract highly specialised parts of the operation to an agent, on the understanding that the latter is better equipped to undertake such specialised activities. The ideal agent should be capable of reporting progress, able to achieve results in a reasonable time, preferably cheaply, and should inform of completion. In the event of failure, which should only occur after all avenues have been explored, a complete explanation should be offered.

The mechanism upon which the experimental file distribution system has been built is based closely on this idea. A set of agent systems with associated private disc storage are available on the network, situated so that there is at least one at each Universe site. Client hosts making use of the system are registered with a particular agent which acts for them when they are involved in transfers, whether as originators or recipients of data. A client system wishing to send a file to another system contacts its local agent, instructs it about the destination and co-operates in transferring the data to the agent. The responsibility for delivery thereafter becomes the agent's. The agent will in general contact the recipient's agent and then, using techniques appropriate for fast transfer over the satellite segment, the data is transferred to the recipient agent. The latter will in due course deliver the file to its client and then notify the originating agent, which in turn informs the original client of the successful transfer. A similar arrangement is used to enable a client to fetch a file from a remote system. In fact the services offered by the agent system are slightly more general: an initiating client may cause a file to be

transferred between two other host systems merely by making a request of its own agent. The agent contacts the agent of one of the hosts involved and the transfer then proceeds much as before. The report on the outcome of the end-to-end transfer between hosts is sent to the client which made the original request. Finally, a client may specify a list of recipient systems for a file transfer.

3. Reliability

There are three aspects of the agent design which contribute towards the goal of reliability. An obvious starting point is that an agent should always be available to a client to accept a commission. In the Universe network, this implies that there is at least one agent situated at each site, since the satellite segment is regularly severed. In general, one agent or more could be situated on each local network to suit local demand. Furthermore, it would generally be arranged for clients to be able to use one of several agents to undertake transfers. This is done, for example, in Grapevine [Birrell81]. However, since this complicates the registration mechanism and also demands a more general way of locating resources, neither of which are the immediate purpose of the current experiment, the facility is not incorporated in the present system. Instead, hosts are registered with a single agent and only this agent can collect from or deliver to its hosts. A consequence of this simplification is that there is a single, global registration of hosts which is replicated in every agent.

It should also be remarked that the agent is conceived as a dedicated system. While nothing in its design or implementation prevents its coexistence with other subsystems, nevertheless in any service environment both performance and reliability militate in favour of dedication.

The aspect of reliability which has received most attention in the design of the agent system is robust communication, both between the client host and its agent and between agents. Typically, file transfer protocols (FTPs) involve the exchange of two distinct forms of information: protocol control messages and bulk data. Procedures to ensure that parts of the latter are not lost or disordered are well known; they generally involve sequencing and retransmission and are often incorporated into one or more of the classes of network service offered by the network. Also included (though not in the network service) is some method of signalling the end of the data, so that the recipient knows whether it has successfully received everything. A variety of protocols which adequately provide these facilities for bulk transfer are available to the Universe agents.

One of the functions of the control messages of a conventional FTP (NIFTP [DCPU81a] for example) is to establish the identity of the transfer at the beginning, whether it be an entirely new transfer or a retry of a previously failed transfer. Another function is to agree which end has responsibility for the file at the end of the transfer, successful or otherwise. The need for this becomes particularly obvious when further activities depend upon the transfer. Failure of either of these functions will lead to either loss or duplication of the file. In the agent design, this aspect of control has been removed from whatever FTP is in use and is instead carried in separate Universe network service level transactions. The particular Universe network service chosen for this is the Single Shot Protocol (SSP). It is a property of the Universe network that the probability of receiving either an incomplete block or a block with incorrect contents is negligeably small. This is not only theoretically true but is also borne out in practice. However, it is perfectly possible for a block to be lost, particularly if there is congestion in an intervening bridge. The strategy adopted is to ensure that all control SSPs are idempotent, that is, they may be repeated and the overall effect is the same as a single successful instance. Obviously, if an SSP request is lost, there will be no reply from the service and the originator times out and repeats the request. However, the originator cannot distinguish this case from the SSP reply being lost and it is in this case that the idempotency is required.

An example of the technique occurs when a client host establishes a new commission with its agent. Commissions are identified within the issuing agent by the assignment of a sequence number. They are identified globally among the set of all agent commissions by being tagged with the issuing agent's identity. The host, of course, requires both acknowledgement of the commission and knowledge of its identity. The initial SSP to an agent contains all the details of the desired transfer; the reply contains the identity. However, at this stage the commission is held and will not be executed by the agent. A further SSP is required from the client, quoting the commission identity, to release it. If the reply to the first of these is lost, the commission can never be released and the agent simply times it out and deletes it; the client, of course, also times out and repeats the request, establishing a commission with a different identity. If the reply to the releasing SSP is lost, it will be repeated until a reply is received; since releasing a released commission is evidently harmless, this SSP is trivially idempotent.

The third aspect of the design has been the pursuit of simplicity, on the elementary premise that the simpler the system the less there is to go wrong. The very idea that the agent should be a separate service on the local network is in no small part so that it can be programmmed specifically for this purpose, with no compromise or generality introduced for some other, conflicting purpose. The idempotent SSP is another example of a technique which not only takes advantage of the properties of the Universe network but is also particularly simple to use,

largely because it is a sequential operation, essentially like an elementary remote procedure call.

4. Design of experimental system

A prototype file distribution system has been built and operated over the Universe network. Since it is of importance to the project to explore techniques for reliable, fast point to point and broadcast transfer over the satellite segment, these aspects of the system have received particular attention. In contrast, the current system does not incorporate authorisation, nor has particular attention been paid to mechanisms for managing registration, distribution lists or access control. The view taken is that while these are obviously important for a service system, a simple scheme will suffice to allow the experimental system to operate and it is not the purpose of the current experiment to explore these aspects. They have been explored in detail elsewhere, notably in Grapevine [Birrell81].

A consequence of the decision to separate control and scheduling interactions from the process of bulk data transfer is that it is simple to incorporate a variety of different mechanisms for collection and delivery of bulk data.

4.1 Client-agent interface

The client-agent interface falls into three parts corresponding to the three activities in which the client hosts participate. The requesting client instructs the agent about the details of the commission and receives a final report on the outcome. This, or indeed any other, client system may also enquire at any time about the progress of the commission. This set of interactions is conveniently grouped together under the title of command interface. The other two interfaces are those invoked during collection and delivery of a file.

(i) Command interface

This consists of a set of SSPs. The mechanism for establishing a commission has already been described in section 3. The final report is conveyed by an SSP which is repeated by the agent until it receives an acknowledging reply from the client. Only after this does the agent delete its spool entry for the commission. A request for status information is carried in an SSP request and the reply contains the answer. This SSP is obviously repeatable in case the reply is lost, on the assumption that what is required is the latest state rather than that at a predetermined instant.

(ii) Collection

This activity is intiated by the collecting agent. The action is one of copying the file: there is no requirement by the agent for the source to delete the file or perform any other action after the copy has been taken and so the agent need not inform the source when it has read the file. An application requiring such action upon successful delivery would make arrangements outside the service provided by the agent.

Thus, for collection, no control messages are required. The agent simply uses whatever procedure is defined for the service to obtain its copy of the file. The procedure may incorporate recovery mechanisms but these will be transparent to the agent. If the procedure reports failure of a temporary nature, the agent will retry from the beginning.

(iii) Delivery

The normal sequence of events during delivery begins with the agent notifying the destination host that a file is available. The host then transfers a copy to itself and notifies the agent upon completion. Both notifications are achieved by idempotent SSPs. The notification by the host that it now has a copy of the file causes the agent to delete the file from its own storage and to begin the process of notifying the requesting client of the successful outcome.

In the Universe environment it happens that there are several desirable destination services (notably print servers) using procedures with the general properties that they are initiated by the sender and they incorporate satisfactory methods of notifying the sender of the outcome of the transfer. Knowledge of how to use this class of procedures was incorporated very easily and opened up a range of services available via the agent without the need to alter those services at all.

4.2 Agent-agent interactions

The most general set of interactions among agents arises in the case of "3rd party" transfer; this is shown in diagram 1. Client host A requests its agent to arrange the transfer of a file between hosts B and C and the agents for A, B and C are all distinct. Agents use idempotent SSPs to pass the commission to another agent for the next stage of processing: typically, A's agent will pass the request to the collecting agent. Once collection is complete, the collecting agent passes the commission to the delivery agent. However, it does not drop out of the chain. The report on the outcome of the transfer passes back along the chain (using idempotent SSPs as usual), triggering deletion of the commission records. The forward links in the chain are used to obtain the latest status of a commission. If intervening agents were to drop out of this chain, additional interactions would be needed to maintain its integrity.

In addition to the passing of commissions, there is also the transfer of the file between the collection and delivery agents. Two procedures are available, point to point and broadcast. The design of conventional protocol techniques for achieving bulk data transfer across a network is

based to a considerable extent on experience with networks whose fundamental transmission properties and switching architecture are considerably different from those being examined in Universe. In previous definitions two principles have tended to dominate. The first is the constraint to operate above a service interface whose suitability for the purpose at hand is open to question. The second is a pretence that a single protocol may be able to embrace a multitude of superficially similar activities; examples are disc to disc transfer, remote printing, data acquisition, teletex etc. A common facet of both of these tendencies is that the bulk data protocol knows nothing of the properties of the underlying network operation and is supposed to be semi-ignorant of the precise nature of the data transfer it is undertaking. In the context of the current project it has been possible to relax both of these principles in the interests of exploring techniques suited to the task at hand.

Two network properties have been taken into account explicitly: the ability to broadcast on the satellite segment and the long round trip delay. The important aspect of the files being transferred between agents is that at both source and destination they reside on direct access disc storage.

4.3 Point to point transfers

The collection agent employs idempotent SSPs to inform the delivery agent of the overall specification of the commission, that the file is now available and what its spool name is. Responsibility for obtaining a copy of the file now rests with the delivery agent.

During the transfer, the sender plays an entirely passive role. An Open Connection is established by the receiver which then sends a series of requests for portions of the file. The sender sends whatever is demanded as fast as possible: it is the responsibility of the receiver to demand only what it anticipates being able to take. The sender maintains no state about the progress of the transfer.

Each portion of the file requested is identified by its size and position in the file. Errors amount to all or part of a portion being lost. The size of a portion is a compromise between keeping as much data "in flight" as possible and maintaining a high probability of the portion arriving correctly without retry, even when congestion occurs in the satellite bridge.

Some preliminary measurements have been made, using (a) the simplest strategy of waiting for reception of one portion to complete before requesting the next and (b) a strategy in which

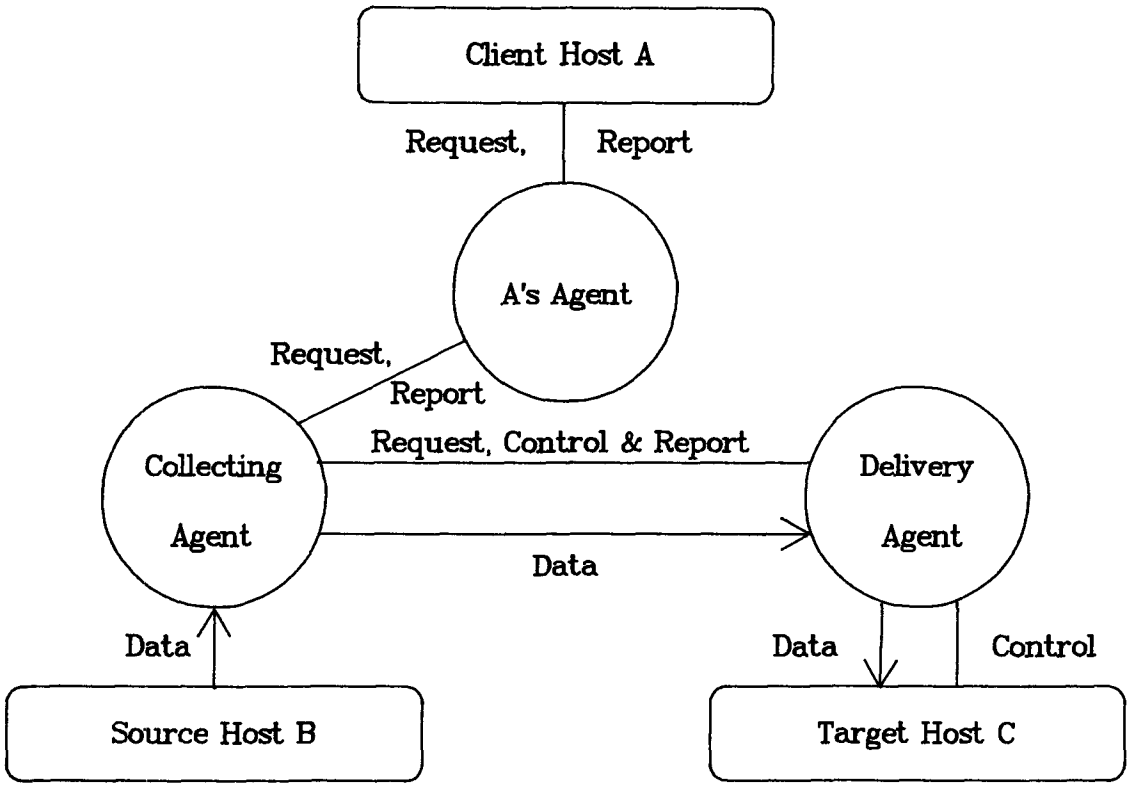

Diagram 1

portions are effectively double buffered. Although the maximum network access rate which the systems used for the experiment can support is about 370 Kbits/s, the maximum point to point rate for the satellite bridge is about 270 Kbits/s [Waters84b]. Using strategy (a) and portions of 100 to 120 Kbytes, overall transfer rates between 190 and 195 Kbits/s were achieved; these are to be compared with the theoretical limit of about 220 Kbits/s. Using strategy (b) and portions of 24 Kbytes, an overall transfer rate of about 220 Kbits/s was achieved, compared with the theoretical figure of about 240 Kbits/s.

These measurements suggest that as the network access speed rises, the size of the portion needed to achieve good throughput using strategy (a) becomes prohibitively high, particularly for the satellite bridge: it may become the principal cause of congestion and therefore run the risk of being partially discarded. Instead, one may use strategy (b) and use the random access nature of both the source and target files to aid recovery of any earlier, corrupt portion. It is hoped that it may be possible to confirm these conjectures and incorporate a robust recovery scheme later in the year using a new satellite bridge and a new satellite.

4.4 Broadcast transfers

Full details of the broadcast file transmission mechanism are given in [Waters84a]; only a very brief outline is included here. Generally, a client host has specified that there is not one but a list of recipients for a file. The collection agent, having obtained the file, examines this list to determine the set of recipient agents. Each of these is contacted by SSP and given a sub-commission whose distribution list is a sub-list of the original. In this case, the initiative for further action remains with the collection agent, which starts up the broadcast file transfer process. At the end of the broadcast, the recipient agents are polled to confirm successful reception of the file. Failures result in a further transfer to the relevant agents; this may be point to point or broadcast, if more than one agent reported failure. The collection agent, having successfully sent the file to all delivery agents, then awaits their reports on delivery. Only when it has all of these does it itself report.

The general principles of the broadcast file transfer are that the transmitter listens for requests from each recipient for sections of the file. A pointer is advanced through the file by the transmitter to record what has been sent. Requests for parts of the file which have already been sent are serviced immediately. Requests for parts not yet sent are delayed by an amount proportional to the number of recipients yet to make a request. At the end of the delay, the section of the file from the pointer up to and including the largest request is sent and the pointer advanced. The process is repeated until the end of the file is reached.

The main objective of the broadcast mode of transfer is to optimise use of the satellite channel; that the algorithm used is successful in this endeavour is confirmed by the transfer time being substantially independent of the number of receiving sites. For further details of broadcast transfer the reader is referred to [Waters84a].

5. Implementation details

5.1 System hardware

The experimental agent is implemented on a 68000 system having 512 Kbytes of main memory and 10 Mbyte Winchester disc backing storage. A 9.6 Kbit/sec asynchronous port is available for connecting a VDU keyboard terminal. The system has an integral ring connection; support for Basic Block Protocol [Leslie84], booting and low level debugging primitives are provided by a 6809 based cpu board having 64 Kbytes of private memory as well as dma access to the 68000 memory. Firmware for this board is contained in a 4 Kbyte eprom. At least one such system (the so-called "Large Server") is available at every Universe site.

5.2 Operating system

The operating system used is Tripos [Richards79], developed at the University of Cambridge, England, which contains in a convenient form all the essential primitives required for a dedicated, special purpose system. It is particularly suitable for network systems development, having a simple, dynamic multi-tasking structure incorporating message passing, coroutine support [Moody80] and a well developed set of supporting tasks and device drivers for the protocols in use in Universe. All programming for the agent system is in BCPL, the language integral to Tripos.

5.3 System details

The agent consists of a number of co-operating tasks, which communicate by exchanging message packets. A Spooler task provides sequenced access to a set of spool entries, each representing an agent commission. Entries are either free, active or on a timer queue; they become active as a result of timing out or external events. Active entries are passed to the Executive task, where the main agent logic is implemented. This task is responsible for generating control SSPs to hosts and other agents. It also manages the loading and execution of bulk transfer code implementing collection from and delivery to hosts, and transfer between agents. The responding end of any transfer is handled by a permanently listening task which manages the loading and execution of code according to information in the initial request block. The structure is completed by a (small) number of identical tasks which handle incoming SSPs from hosts and other agents. As well

as handling control messages, these tasks implement remote diagnostic and debugging aids for the agent.

5.4 Agent configuration and host registration

Two files provide the information needed by the agent system to enable it to decide which agent to contact for collection or delivery of a file. The set of agents co-operating to provide the service is defined in a file called the agent configuration file. Each entry in the file binds the Universe global name of an agent system to a unique integer, its identity. This identity has two uses: as a constituent of a commission identity and as the permanent identifier associated with the contents of the particular agent's spool storage. The latter is established at cold start, when the spool is created, and is checked upon the occasions of subsequent warm starts. This is the only place where the identity of an agent is bound to its network name. Adding or removing an agent from the system is essentially a matter of inserting or deleting the corresponding entry from this file. While adding an agent may be done at any time, in the prototype system agents should only be removed when the system as a whole is quiescent and the agent in question is empty. The indirection provided by the configuration file also allows the possibility, in a service environment with mountable discs, of moving a particular agent to a different machine in case of, for example, machine failure.

A second file, the host directory, serves several functions, the primary one being to register with an agent every host or service participating in the file distribution system. Strictly, a name registered in the directory is a partial or complete Universe network name of a service available on a participating host The names are public, in that a client host uses them to specify the acquisition and delivery services to be used at the end points of a file transfer. The agents use the directory to determine which agent should handle the acquisition or delivery of a file. Also recorded against each name are the collection and delivery procedures to be used. In some cases, one of these procedures is null because it only supports either collection or delivery. Examples are data acquisition systems and print servers. The procedure identity is bound by table lookup in an intermediate file to the name of a file containing the procedure code. The arrangement has been found very convenient experimentally, since the addition of a particular procedure may in many cases be done simply by transferring the necessary code to support that procedure and creating a new intermediate lookup file. The procedure will then come into use as soon as a host is registered as using it. The service end of a procedure for file transfer is managed in a similar fashion. Addition or removal of a host service from participation in the agent system is essentially a matter of inserting or deleting the relevant entry in the host directory.

Both the agent configuration file and the host directory are global. Since every agent obviously requires access, they are replicated in full in every agent.

6. Review and applications

In building the prototype system, some issues have been avoided and for others temporary expedients have been employed. One such issue is the interpretation of a file being transferred. It is a principle of the system that the agents need not be concerned with this: a file is merely an ordered set of bits. While such interpretation should remain an end-to-end matter, it is currently an open question whether conveying the information to enable interpretation should remain entirely outside the scope of the system or be carried in a part of the commission specification not interpreted by the agent.

Another issue requiring resolution for a production system is that of maintaining the registration and configuration databases. Both of these are currently replicated in every agent and, while this ensures their availability, it raises questions about maintenance and consistency. One interesting possibility is to make use of the bandwidth of the network to distribute complete new versions of the database, rather than a sequence of updates. A related question requiring examination is the toleration of the system to the databases being temporarily out of step.

A third issue is that of scaling. The current system is designed having in mind a maximum file size of some 1-2 Mbytes and a maximum of 100 spool entries available on each agent for recording commissions (though these limits could not both be met simultaneously). While adequacy of these limits can only be reliably determined in the context of a particular application, it is anticipated that in many cases the limit of 100 spool entries may be too small. One solution is to increase the number of agents; another is to redesign the spool system to have a larger capacity. Neither is a change to the system architecture.

6.1 Applications and extensions

The file distribution system was conceived with several candidate applications in mind. Within the Universe project there is a requirement to distribute software, ranging from the Tripos operating system for 68000s to dedicated code used in Z80s to control speech access to the network. Provision of printing services of various degrees of sophistication could obviously be based with advantage on such a system. A particular form of this, geared to document distribution would be valuable to any project whose participants are widely dispersed. Since the agent is not concerned with the content of files, the same system could be used equally well for facsimile document transfer as computer or wordprocessor generated

generated documents. A particularly appealing direction for development is that of mixed media document transfer, where documents may consist of computer or facsimile text, bit-map or graphic pictures and voicegrams.

The activities of the agent system have features in common with a distributed electronic mail system, and it is to be expected that an implementation of the latter could be based on the former. Another possible area for application is to distributed job transfer systems of the sort described in [DCPU81b], [ISO83a], [ISO83b]. Some of the principal features required of such a system are already present; the main investigation would centre on ways of incorporating an execution host (or "job mill" in JTM terminology), consistent with the acquisition and disposition hosts already present.

Currently the service offered is a store and forward one at the granularity of a file. Voicegrams and bit-map images are typically much larger than text documents or programs. A possible extension to the architecture which might be stimulated as a result is the management of file transfer by reference, rather than by value as at present. Two architectural changes would increase reliability further. The first would allow a client to be registered with more than one agent. The second would incorporate into the system the knowledge of the existence of X25 links between remote sites and how to use these when the satellite is unavailable.

Acknowledgements

I should like to extend here my thanks to the many colleagues in the Universe project who have contributed to the development of the system described here. In particular, Roger Needham first suggested the idea and together with John Burren contributed many of the ideas on point to point satellite transfer. I am indebted to Ian Leslie who developed the broadcast file transfer facility used by the agents, and also to Steve Wilbur, Nick Pope, Bill Tuck and Gerry Morrow who contributed through many helpful discussions to the eventual design.

References

Adams82 G C Adams, J W Burren, C S Cooper, P M Girard: "The interconnection of local area networks via a satellite network", New Advances in Distributed Computer Systems, K G Beauchamp (ed.), pp 201-210 (1982).

Birrell81 A D Birrell, R Levin, R M Needham, M D Schroeder: "Grapevine: an exercise in distributed computing", Commun. ACM vol.25, no.4 pp 260-274 (April 1982).

DCPU81a Data Communications Protocol Unit of DoI (UK): "A Network Independent File Transfer Protocol" (1981).

DCPU81b Data Communications Protocol Unit of DoI (UK): "A Network Independent Job Transfer and Manipulation Protocol" (1981).

ISO83a "Job Transfer and Manipulation Concepts and Services", ISO working document, ISO/TC97/SC16 N1458.

ISO83b "Job Transfer and Manipulation Protocol", ISO working document, ISO/TC97/SC16 N1248.

Kirstein82 P T Kirstein, J W Burren, R Daniel, J W R Griffiths, D King, C McDowell, R M Needham: "The UNIVERSE project", ICCC'82 pp 443-447 (1982).

Leslie83 I M Leslie: "Extending the Local Area Network", PhD thesis (1983); available as Technical Report No.43 from Computer Laboratory, University of Cambridge, England.

Leslie84 I M Leslie, R M Needham, J W Burren, G C Adams: "The architecture of the Universe Network", ACM SIGCOMM '84 Symposium on Communications Architectures and Protocols, Montreal.

Moody80 K Moody, M Richards: "A Coroutine Mechanism for BCPL", Software - Practice and Experience vol.10 pp 765-770 (1980).

Richards79 M Richards, A R Aylward, P B Bond, R D Evans, B J Knight: "TRIPOS - A portable operating system for mini-computers", Software - Practice and Experience vol.9 no.7 pp 513-526 (July 1979).

Waters84a A G Waters, C J Adams, I M Leslie, R M Needham: "The use of broadcast techniques on the Universe network", ACM SIGCOMM '84 Symposium on Communications Architectures and Protocols, Montreal.

Waters84b A G Waters: "The Performance of the Satellite Bridge in the Universe Project", Symposium on Performance of Computer-Communications Systems, Zurich (March 1984).