A Java-Based Framework For Comparing Mobility Policies In Networks With Dynamic Resource Management

Marc Boissaux[†], Alistair Munro[†], Ian Marshall[‡], Paul McKee[‡]

[†]Centre for Communications Research, University of Bristol, Bristol BS8 1UB, UK, {Marc.Boissaux@bt.com, A.Munro@bristol.ac.uk}, Tel +44 (0)1473 647 668, Fax +44 (0)1473 643 345

[‡] BT Labs, Martlesham Heath, Ipswich IP5 3RE, UK, {Ian.W.Marshall@bt.com, Paul.McKee@bt.com}

Abstract

Next-generation multiservice networks are expected to offer object mobility support as well as node programmability. This paper describes how this may be used to extend the current vision of policy-based network management. It then proposes a system intended to enable evaluation of different mobility policies. We introduce the notion of a network centre of gravity, and use the evaluation framework to show that policies based on it can be advantageous.

1.Introduction.

Today's communication networks are becoming capable of providing more and more diverse services as the traditional distinction between communications systems and information technology is disappearing. The continuing extension of user services can only be maintained by improving the use of resources by the network and by the elements connected to it. One of these structural improvements centres on the provision of network mobility, which we use in this paper to mean the ability to move between locations in the network. While this is commonplace for users of network services today, it is also somewhat limited. New applications will require that

1) the mobility concept is extended to encompass system and application objects as well as network users;
2) the network becomes autonomously reconfigurable, or programmable, to a much greater degree than currently accepted.

Networks that can meet these requirements are still largely at the specification stage, and there are conflicting views (e.g. active networks [1] and OPENSIG [2]) on how they should be implemented. Some convergence of positions can already be seen [3]. Furthermore, current active network projects (e.g. [4], [5], [6]) aim to optimise the quality of existing network management.
Once both the above requirements are met, progress can be made with respect to the concept of network management itself. The possibility of continuous configuration, or management, of object location is opened up. When individual nodes in a network can be reconfigured on the fly, object location can be controlled such that it responds to changing network conditions. The most straightforward approach to mobility control involves the use of *policies*. By providing appropriate object mobility policies within a framework that supports them, it will be possible to extend the domain of what is currently understood as policy-based network management[1] to encompass aspects of automatic traffic optimisation as described in the following paragraph.
Within current networks, the amount and shape of traffic generated by distributed applications cannot be optimised at the object level. The reason for this is that the smallest atomic unit managed when working with stationary entities is the single object. Management of traffic can thus only be carried out through managing the interactions between multiple objects. On the other hand, introducing mobility into the picture offers the interesting possibility to optimise the management of single objects by evaluating the trade-off between

1) a remote transaction generating a *transaction load* on the network;
2) object movement followed by a local transaction, which generates a *migration load* on the network.

It is possible to envision a network wherein a range of policies governing the mobility of an object are stored, each adapted so as to optimise the above trade-off for a certain combination of application, user, or load types. Such policies may then be executed dynamically using the network's programmability

[1] The state of the art in policy-based network management research, as exemplified in [7], largely emphasises the *authorisation* and *obligation* aspects of network management.

features. In this way, the advent of programmable next-generation networks with object mobility allows for the refinement of network management.

The purpose of the work described in this paper is to specify the structure of a system which could be used to optimise this trade-off.[2] Section 2 gives a high-level view of the prototype system implemented. Section 3 introduces the notion of a network centre of gravity, and justifies why this is thought to be a powerful and flexible concept for approaching the optimisation problem described. Section 4 lists a series of factors which were identified during system testing as having an impact on system performance, and must thus be taken into account when optimising policies are developed. Section 5 shows the results of a simulation of a scaled-up version of the initial implementation.

2. High-level view of the system.

As a feasibility study, we have implemented a mobile mail service. It involves the following three main types of objects:

1) The *user object*, which is mobile and follows a predefined pattern in order to model a human user moving through the network and accessing the mailbox service at various points. Every user has a personal mailbox and is able to read the contents of their own mailbox as well as write to any mailbox in the system. User moves in-between requests, as well as request patterns (inter-request deltas, read/write distribution) and distribution of request sizes, are taken from actual mail traces. The user choice of desired mailbox location for service transaction is based on the quality of service (QoS) they expect from different locations.
2) The mobile *mailbox object* contains user mail, and moving it across the network generates a certain load. It can send any of its mail messages to the user on request. The mailbox also changes size as messages are received, such that its (migration load / transaction load) ratio evolves in time.
3) The stationary *mail server object*. Every system-related object hosted at a mail server location, whether user or mailbox, represents a processing and network load. Hence, when a user finds that the QoS expected from the mail server currently hosting the mailbox of interest is too low, and suggests an alternative mail server, that server may refuse this request if it is unable, or unwilling, to currently fulfil it. Thus, both user and mail server are involved in the mailbox (i.e. service) movement decision.

[2] Hamilton and Mitrani [8] tackle a similar problem from an analytical viewpoint.

A further object of major significance is a *traffic generator* running on a single node in the network. For resource availability reasons, the test implementation was only deployed within a LAN. In order to approximate the proposed WAN as closely as possible, and to provide a mechanism that allows for scalability of background traffic, the traffic generator simulates the distribution of WAN-type traffic within the implementation LAN.
The data series used for traffic modelling consists of two weeks' worth of SMTP traffic statistics, both incoming and outgoing, collected from the University of Bristol servers. Data features modelled are the time between requests, the byte length of each request and the decision whether successive writes should be made to the same destination or not. User objects replay single-user traces filtered out of the data in order to generate accurate user loads.
The mobile code functionality is provided by IBM's Aglets platform [9].
The system also includes a *policy interpretation mechanism* intended to support automatic policy management for optimisation purposes. A simple policy definition language, or PDL, was defined supporting the following rule types:

- The `POLICY` rule gives the name of the policy as well as the type of object (user, mailserver, mailbox) it applies to.
- The `USES` rule gives names of resource types involved in carrying out policy decisions in case resource checking has to take this into account.
- The `CONSTRAINT` rule gives a Boolean-compatible expression which will be evaluated when a policy decision is made.
- The `ACTION` rule describes the action(s) to be taken in case the constraint is verified.
- The `ALTERNATIVE` rule describes action(s) to be taken in case the constraint is not verified.

It is thus possible to make runtime decisions on which alternative functionality should be applied depending on constraints. Hence, the PDL is thought to offer the flexibility required by its purpose while remaining as simple as possible.

3. The notion of a network centre of gravity.

The figure below shows the suggested goal of policy-based optimisation.

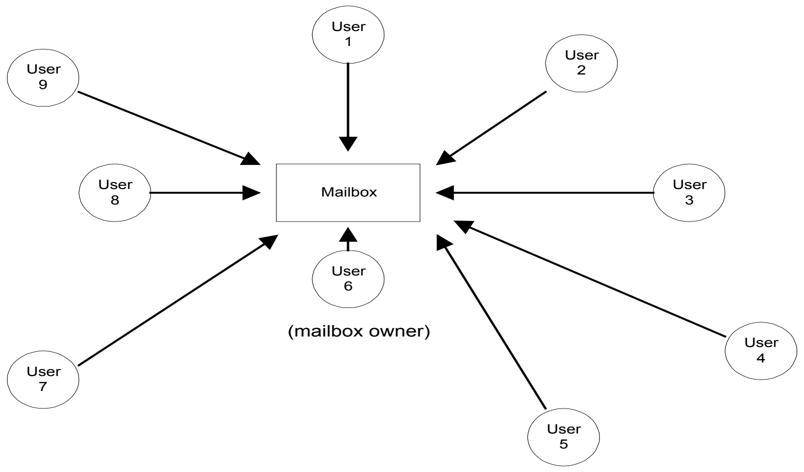

Fig. 1. The network centre of gravity.

Between the two extremes of keeping the mailbox stationary and having it move precisely to wherever a user is calling it from, there is a continuum of possible policies regarding service movement. It is necessary to define a concept by which this continuum can be described. The proposed concept is that the 'best' service location has to be analogous to a *centre of gravity*. As the figure shows, a mailbox will be used by a number of users, all of which, except for the owner, will merely write to it. Let the arrow lengths in the figure represent a valid distance metric.[3] Since all users will be trying to make the mailbox move to a location as close to their own as possible, they can be visualised as "pulling" the mailbox toward them, with the strength of the attracting vector proportional to the number and frequency of requests made. This means that an optimal compromise for the service position would formally correspond to a centre of gravity where individual user weightings are determined by their frequency of requests. Thus, in the figure, the mailbox owner, as the person with (normally) the highest frequency of requests, will have the strongest weighting and the mailbox will, on average, be closest to

[3] Such a metric should reflect capacity as well as current loading for both network link and local machine.

that user. Clearly, the position will fluctuate; the important point is that, given regular user habits and a regular updating of the statistics involved, it *will* be possible to define a centre of gravity area in a meaningful manner.
It should be noted that a network centre of gravity is thus specific to every single service instance and dependent on request statistics unique to that specific entity. This approach is believed to be superior in two important ways:

- The load is naturally balanced over the network as a result of the creation of separate centres of gravity. The quality of the balancing is expected to improve with the number of entities in the system. We expect that load balancing can thus be provided on top of transparent QoS optimisation.
- Services are not always accessed by the entirety of users in a network. This is particularly clear in the case of a mail service, where users by and large tend to send mails to a reasonably restricted, and closed, set of recipients only. Hence the keeping of separate sets of statistics for each service instance is believed to result in a centre calculation which is much more precise and adapted to the actual load. In a general set of statistics, all users unknown to the current user would function as noise and degrade the quality of the statistics. In more general terms, the use of multiple centres of gravity brings about a much finer *granularity* with respect to the load data. As an example, a current area of research (e.g. [10][11]) involves the identification, and description, of separate regional types of user behaviour regarding their use of the Internet. Once such types have been identified, they may be associated to mobility policies each covering the particular mobile service to which the type applies. If a user is found to approximate one of the types, the appropriate policy may be triggered. Similarly, if a user's mailbox is found to be accessed by a "strong" set of recurring access habits, its centre of gravity recalculation interval may be increased, or its collection interval for statistics lengthened. In that sense, fine granularity of data provides further opportunity for resource usage optimisation.

4. Factors influencing system performance.

The prototype system implementation has successfully undergone validation testing. As a result we identified several factors all impacting on system performance:

- Number of nodes in the network. Clearly, a certain number of nodes must be available in order to allow for sufficient differentiation of mobility policies.

- Heterogeneity of the network. The nodes which make up a network will run different operating systems, feature different hardware architectures, and deliver widely different performance levels. In a network which only contains a small number of nodes, atypical machines entirely dominate the descriptive statistics generated. For instance, in a network which only contains a few machines, a single fast machine will nearly always feature the fastest response times, such that users pursuing a high perceived QoS will request it again and again. With many machines, on the other hand, averaging takes place and performances that differ significantly from the mean will not unduly affect results.
- Number of objects (i.e. users and mailboxes) in the system. In addition to the above point, user request patterns are applied at random, such that user behaviour only gets averaged sufficiently if users are present in sufficient number.
- Average size of service objects (i.e. mailboxes). As a mailbox grows bigger, the migration load it generates when moving across the network increases. At the same time, however, the service load caused by transactions remains constant, since the nature and topology of mailbox access is in no way linked to the current size of the mailbox. Hence an increase in the average mailbox size will increase the ratio between migration load and transaction load. This is expected to shift the trade-off balance away from mobility.
- Topology, capacity and loading of the tested network also have a significant impact on system performance, since they determine the maximum throughput available at any one point in time. For typical measurement intervals, it was around 200 kilobytes per second.
- More generally, any factors pertaining to the topology of the setup used for system implementation contribute to the output of the system. Thus, internal Aglets signalling, the performance characteristics of the Java network I/O library used, the class loading policy implemented by the framework ([12], p.165) or similar factors are all reflected in the traffic produced.

Most of these factors will become less and less significant due to averaging as network size is increased, such that they become immaterial in real-world networks. However, resources available for testing were insufficient to offset these factors. Hence a simple simulation was built and tested in order to validate the centre of gravity concept. This is described in the next section.

5. System simulation and results.

The simulation uses a two-dimensional grid of points, over which users move according to predetermined patterns that are random but fixed through reuse of the same randomly generated files. The distance in squares is taken as a meaningful network distance metric. Users randomly choose mailboxes to make requests to; however, the probability is weighted such that a user's own mailbox is queried preferentially. Mailboxes then decide whether to move, and where to move, before performing the request accordingly to their current policy. The postulation of a network distance allows for evaluation of user latency (transaction distance, related to transaction cost) and the number of mobility rule firings (incremented if the service moves and thus related to migration cost)[4].
Mailbox policies used correspond to

- an immobile mailbox,
- a mailbox which always moves to the user location, and
- three different approaches to the centre of gravity.

All three centre of gravity approaches involve an object which is attached to each mailbox and keeps a list of all requested locations ordered by the number of times each has been requested. When a request is received, the request destination's popularity is checked against the list, and the request refused if the destination is not popular enough. In this sense, a classification of destinations is made which, although it does not actively determine a centre of gravity, nevertheless rules out locations unfit for this role.

- The first approach ("Simple") involves a simple binary decision made subsequently to checking the requested location's suitability as a destination. If the location is found acceptable, the mailbox will move there; if it is not, it will remain stationary and remotely fulfil the service request.
- The second approach ("Seeker") attempts to find compromise locations. It tests the initial requested location as before, but goes on to evaluate points at an increasing distance from the user for suitability. It always tests an entire orbital of points equidistant from the user, and returns the one that, on the current orbital, is closest to the mailbox. Since points tested become more and more distant from the user, the first acceptable point found is the

[4] It is suggested that a realistic migration cost metric would be of the type ax+b, where x is the distance of migration. b denotes a fixed cost indifferent of migration distance. The 'rule firings' metric gives that fixed cost, and is thought to give a good initial approximation to migration load.

most desirable one, and is therefore used. If no points closer than the current mailbox location are found, that location is returned.

For these two approaches, the actual acceptance/refusal cut-off point, given by a ratio (between 0 and 1) showing the proportion of points in the network that have to be less popular for the current point to be acceptable, is clearly an important algorithm parameter. A higher value means that more requests will be refused, and that the service object is more likely to transact the service from its current position. Conversely, the lower the cut-off value, the greater the proportion of acceptance of the initial movement request. In this sense, centre of gravity implementations of the proposed type allow one to optimise the compromise between the 'unconditional mobility' and 'stationary object' approaches.
However, the two approaches given so far do not actively target the centre of gravity, but instead attempt to eliminate proposed points which clearly do not correspond to it. The third approach proposed ("Active") entirely neglects the user's requested location, and evaluates the *n* most popular locations instead, returning the one for which the sum of distances from user and service is smallest. Since the most popular locations are chosen specifically, the cut-off point chosen is not expected to be of much relevance.

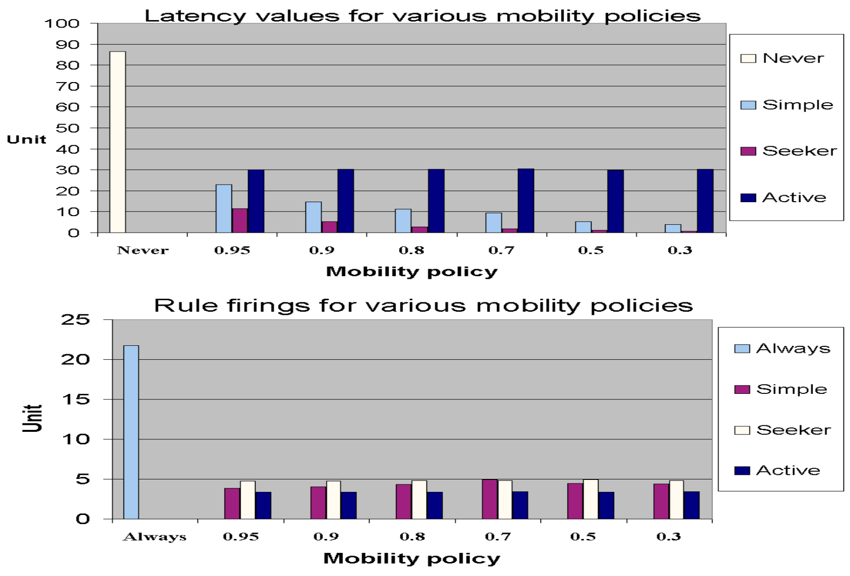

Fig.2. Simulation results.

These expectations are confirmed by the graphs given as Figure 2. Entries on the x-axis correspond to the stationary object policy ("Never", giving a zero value for migration cost), the unconditional mobility policy ("Always", giving a zero value for latency) and the three centre of gravity policies evaluated for the different cut-off points given.[5] The graphs were obtained by running five separate sets of 100 users making 300 requests each. Individual user results were blended to give overall results with respect to a standard interval length of 100 ms. The five separate series were then averaged in order to improve statistical quality.

The different centre of gravity approaches can be seen to offer the expected compromise between stationary objects (which do not generate any migration cost, but cause large latency values and transaction load) and unconditionally mobile objects (which optimise latency at the cost of migration). More significantly still, the three approaches can be seen to offer different compromises. The basic, binary-decision strategy ("Simple") reduces migration cost significantly with respect to unconditional mobility while delivering a quality of service (latency from the user point of view) that is very strongly improved from the stationary state. Thus, the latency for a cut-off setting of 0.3 corresponds to a reduction by 95.4%.
The second approach ("Seeker") is even superior to the basic one from the latency point of view, while maintaining a similar level of mobility load. Since its effort is focused on finding a location that is suitable for the current user, it further improves on the transaction cost and latency metrics, while retaining a migration cost metric similar to the basic implementation. It can also be noted that, for both these policies, decreasing the cut-off point (i.e. making the mailbox more likely to accept moves) decreases the latency obtained. On the other hand, the impact on migration cost is only slight.
The third implementation ("Active") represents a different set of priorities. Reduction of service movement is made more significant than maximisation of user-perceived quality of service. Hence, unsurprisingly, user latency has gone up while the generated migration load has, on average, gone down.
The three approaches to the centre of gravity given are tentative, but clearly prove the flexibility of the concept. Different implementations will be able to compromise between migration-related load and transaction-related load in different ways. In this way a linear optimisation problem has been defined, within which the triggering of different policies allows for continuous adaptation.

[5] For example, a value of 0.95 means that only points in the top 5% of the ordered request list are seen as acceptable.

6. Conclusion.

We believe that active networks, or a similar next-generation network technology offering support of both programmability and object mobility, offer significant advantages in the area of network management. Existing projects have used the flexibility of programmable networks to enhance current network management functionality. We suggest that, within the more advanced technological context given by active networks, the scope of network management itself may be extended in a natural way to incorporate automatic optimisation management of mobile objects. Such management can be carried out using policies adapted to the management situation and the managed entities, possibly downloaded to active nodes executing services, and executed on the nodes.

The present paper sketches this research context and describes a system which was developed in order to make qualitative evaluation of different mobility policies possible. Validation testing was successfully carried out using an implementation of the system, and gave rise to the identification of a number of factors which will impact on system, and thus policy, performance. The paper also proposes the concept of a network centre of gravity for representing the mobility optimisation problem.

The concept is then justified through a description of simulation results obtained. The three different approaches given show the flexibility of the centre of gravity with respect to mobility management. They identify one parameter, the cut-off point, which allows for adjustment of the centre location, and prove that the concept does allow for a variety of compromises between unconditional mobility and stationary objects.

7. References.

- [1]: http://www.darpa.mil/ito/ResearchAreas/ActiveNetsList.html
- [2]: Opensig site, http://comet.columbia.edu/opensig/.
- [3]: A T Campbell et al. “A Survey Of Programmable Networks”, 1999, available from http://comet.columbia.edu/~campbell/andrew/publications/publications.html.
- [4]: D Raz, Y Shavitt, “An Active Network Approach To Efficient Network Management”, 1st International Working Conference on Active Networks, Berlin 1999, published in Active Networks: Proceedings, Springer Lecture Notes in Computer Science 1653.

- [5]: K Sugauchi et al., "Flexible Network Management Using Active Network Framework", 1st International Working Conference on Active Networks, Berlin 1999, published as above.
- [6]: B Schwartz et al., "Smart Packets For Active Networks", presented at OpenArch, March 1999, available from http://www.net-tech.bbn.com/smtpkts/ smtpkts-index.html
- [7]: M Sloman, E Lupu, "Policy Specification For Programmable Networks", in Active Networks, First International Working Conference, Berlin 1999, Proceedings, Springer Lecture Notes in Computer Science 1653.
- [8]: M D Hamilton, I Mitrani, "Optimal Allocation Policies for Mobile Agents", University of Newcastle (UK), to be submitted.
- [9]: Aglets homepage, http://www.trl.ibm.com/aglets
- [10]: V F Almeida et al., "Analyzing the Behavior of a Proxy Server in Light of Regional and Cultural Issues", 3rd International WWW Caching Workshop, Manchester, UK, 1998, available from http://wwwcache.ja.net/events/workshop/21/.
- [11]: J Pitkow, "Summary of WWW Characterizations", 7th International WWW Conference, 1998, available from http://www7.scu.edu.au/ programme/fullpapers/1877/ com1877.htm.
- [12]: D B Lange, M Oshima, Programming And Deploying Java Mobile Agents With Aglets, Addison-Wesley 1998.