

On the use of Agent Technology for IN Load Control

Åke Arvidsson^a, Brendan Jennings^b, Lars Angelin^c and Magnus Svensson^c

^aEricsson AXE Research and Development, Soft Center, S-372 25 Ronneby, Sweden¹

^bTeltec Ireland, Dublin City University, Dublin 9, Ireland

^cEricsson Software Technology, Research, Box 518, S-371 23 Karlskrona, Sweden

Recent years have seen increases in the number, complexity and usage of telecommunications services, making the task of dimensioning networks to ensure acceptable performance levels increasingly difficult. Load Control strategies play a key role in performance management, thus as networks are evolving the need for efficient and flexible strategies is growing. In this paper we contend that a suitable paradigm for the development of strategies of this type may be provided by Agent Technology. To support this contention we present an agent-based IN load control strategy that utilises a 'Market-Based Control' approach, and results of a simulation study which shows that this strategy performs well under a range of load scenarios.

1. INTRODUCTION

Rapid technological advances have encouraged ever-greater usage of telecommunications services, both in terms of the number of users and the information volumes that they produce. Continued growth in the market is welcome from the point of view of network operators and service providers, however the task of meeting customers' availability and service quality demands is becoming increasingly challenging. Convergence of fixed, mobile and Internet networks, together with the expected introduction [1] of service delivery and management platforms based on distributed object technology is resulting in an increasingly complex, interconnected infrastructure that requires a co-ordinated approach to performance management. A particular difficulty in this changing environment is the dimensioning networks to meet the performance requirements of an, as yet unknown, set of services and their associated usage demands. Because of this difficulty networks are becoming more susceptible to overloads, which result in customer dissatisfaction and, in cases where service requests are abandoned, loss in revenue for the network operator.

Traditional approaches to Intelligent Network (IN) load control have centred on the protection of individual Service Control Points (SCPs); for a discussion and comparison of a number of strategies of varying sophistication the reader is referred to [2]. We argue that agent

¹ The work was carried out on behalf of The Department of Software Engineering and Computer Science, The University of Karlskrona/Ronneby, S-372 25 Ronneby, Sweden.

technology may provide a paradigm that is particularly suited to the realisation of ‘network-based’ [3] IN load control strategies. Network-based load control strategies differ from traditional approaches in that they seek to optimise the usage of the resources of the network as a whole, rather than on a localised basis, thus they appear particularly suited to the types of distributed solutions normally associated with agent-based applications. In this paper we outline a multi-agent architecture for IN load control and an algorithm based on this architecture that employs a ‘Market-based Control’ approach.

2. MOTIVATION FOR ‘NETWORK-BASED’ APPROACH

Existing IN load control strategies are generally ‘node-based’ in nature – operating by throttling service requests in response to detection of SCP overload onset. We argue that, given current and future increases in number and usage of IN services, node-based strategies alone cannot guarantee that desired performance levels are consistently achieved. The following observations support this view:

- By their nature node-based strategies are not able to control traffic in a manner which is optimal from the perspective of the whole network [3]. For example in configurations where SSPs may choose between a number of SCPs for the processing of service requests a set of independent node-based strategies can not ensure that load is balanced across all SCPs. Maximisation of network throughput and load balancing between resources are goals of network-based strategies not addressed by node-based approaches;
- Node-based strategies serve to protect individual nodes only and may indirectly cause the propagation of overload [4], resulting in adverse effects on the service completion rates of the network as a whole;
- Node-based strategies typically focus on SCP protection only, however a network operator will typically also be interested in maximising network revenue and/or the enforcement of service priorities [2, 3];
- Node-based strategies may not interact effectively with the protection mechanisms that are incorporated into the signalling networks that carry information between the IN nodes. This lack of co-ordination means that in some circumstances node-based strategies are at best ineffectual and can even serve to exasperate the overload situation [5];

We contend that agent technology offers the potential to develop ‘network-based’ IN load control approaches that avoid the shortcomings of node-based strategies and that will be flexible enough for application in existing and future IN environments.

3. MOTIVATION FOR USE OF AGENT TECHNOLOGY

Historically the origins of agent technology arose through efforts by researchers to apply techniques from Artificial Intelligence (AI) in a distributed context. This led to the field of Distributed Artificial Intelligence (DAI) which has since evolved into a very broad area of research and commercial work, known as ‘Agent Technology.’ A good overview of the field can be found in [6].

We note that flexible and adaptable network load control strategies could certainly be implemented using standard software engineering techniques, however we believe that there are many advantages to be gained through the adoption of an agent-based approach:

- *Methodology*: the agent paradigm encourages a knowledge/information centred approach to application development, thus it may provide a useful methodology for the development of control strategies that require manipulation of large amounts of data collected throughout a network.
- *Agent Communication Languages*: much work in the agent research community has focused on development of advanced communications languages that, for example, allow the negotiation in advance of the semantics of future communications. This degree of flexibility is not present in traditional communications protocols and would be of use in realising strategies that adapt to dynamic network environments where, for example, traffic patterns change due to introduction/withdrawal of services;
- *Adaptivity*: agents may have adaptive behaviour allowing them to learn about the normal state of the network and better judge their choice of future actions. A substantial body of work relating to learning behaviour in the context of agent systems exists and could be leveraged to incorporate learning behaviour into a load control strategy;
- *Openness*: the agent approach is very open, in that individual agents may exchange data and apply it in different ways to achieve a common goal. This means that equipment manufacturers could develop load control agents tailored to their own equipment, but which could still effectively communicate with agents residing in other equipment types;
- *Scalability*: the agent approach allows for increased scalability to larger networks. For example an agent associated with a recently introduced piece of equipment can easily incorporate itself into the agent community and learn from the other agents the range of parameters that it should use for its load control algorithms;

In summary it can be said that investigation of the use of agent technology for network load control is an attractive research topic as it may provide a useful knowledge-centred methodology for building flexible, adaptable, open and robust solutions for what is an inherently distributed problem.

4. MULTI-AGENT ARCHITECTURE FOR IN LOAD CONTROL

In this section we propose a multi-agent architecture for IN load control, taking as our starting point the observation that IN load control can be seen as a distributed resource allocation problem. On one hand, there is the desire to serve all service requests that are often unpredictable and bursty with regard to time, rate and volume. On the other hand, all the resources in the network have finite capacity and must be managed for optimal allocation amongst different usage requests. Resource allocation in IN involves three tasks: (1) monitoring utilisation levels of resources (in particular SCP processors), (2) control of the allocation of these resources and (3) granting/denial of permission to individual service requests to use the resources.

In our architecture, there are three agent types, 'Allocators,' 'Distributors and 'Quantifiers,' which fulfil these three tasks:

Allocators

Allocators are associated with Service Switching Points (SSPs) that act as the points of entry for service requests. They perform task (3) – controlling the entry of service requests into the network. To do so they form a view of the load situation in the network and the possibility of resource overload, based on their own predictive algorithm(s) and information received from other agents. If they perceive that there is danger of overload of resources they will throttle service requests on a priority basis. Where possible they also control the routing of traffic, with objective of achieving optimal load balancing.

Quantifiers

Quantifiers perform task (1) – the monitoring and prediction of the load/performance of SCPs (and possibly other IN resources) and report this information to other agents. They also implement node-based protection strategies involving the selective discard of traffic in response to local overload onset detection.

Distributors

Distributors maintain an overview of the load and resource status in the entire network, through communication with other agents and use of their own analysis algorithm(s). They perform task (2) – playing a controlling or supervisory role in allocating resources associated with Quantifiers to Allocators.

The three agent types form a hierarchy of competencies in terms of the reasoning capabilities they use to decide upon their actions. This hierarchy could be further extended by Reporter agents operating on a longer time-scale which could employ more ‘AI-like’ algorithms to, for example, provide recommendations on the introduction of additional SCP processing capacity in response to observation of recurring overload conditions.

It is to be noted that while agent-based load control strategies may appear attractive there are many practical issues that must be addressed relating to the introduction of agents into existing telecommunications equipment. A discussion of some specific issues relating to the integration of agents into an IN environment can be found in [7].

5. NETWORK SCENARIO

Figure 1 below presents a simplified IN configuration, containing I SCPs and K SSPs, where I is assumed to be less than K . It is assumed that the SCPs all support the same set of J services classes and all user service requests can be directed by the receiving SSP to any SCP. The signalling network used to transport messages between SSPs and SCPs is represented as a ‘cloud’ rather than any specific topology of signalling links and nodes. All messages passing through the signalling network cloud are assumed to experience one of two delay values (chosen at random), no messages are lost and signalling network resources do not overload. Also shown in Figure 1 are the Allocator and Quantifier agents associated with SSPs and SCPs and a single Distributor agent, whose physical location is unspecified.

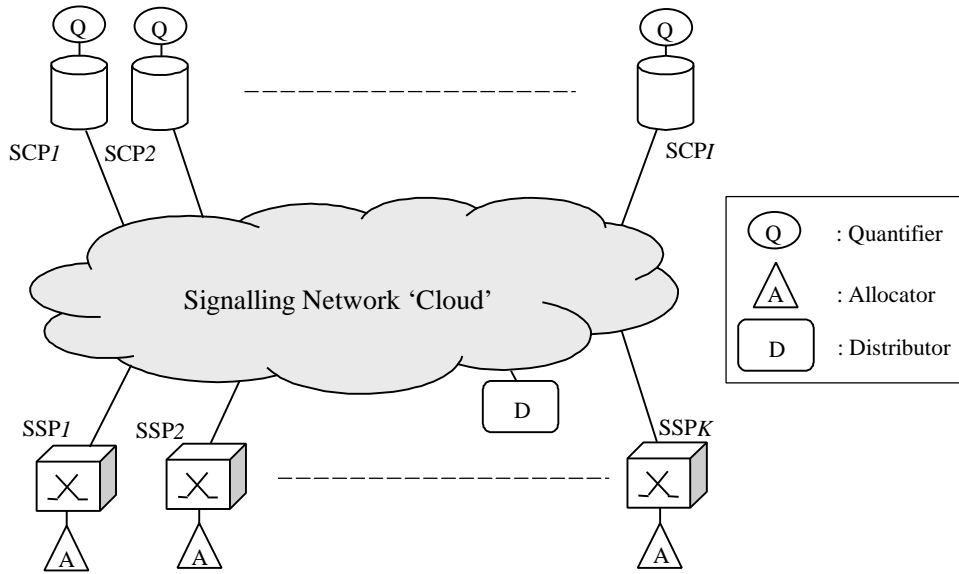


Figure 1: Simplified Intelligent Network Scenario

In Figure 1 only a single Distributor agent is present, due to the fact that in this paper we will discuss a centralised version of a market-based strategy, where the Distributor runs an auction process that allocates resources for the whole network. However, as discussed in §8 distributed versions of the strategy involving multiple concurrent auctions are possible.

6. ‘CO-OPERATIVE MARKET’ STRATEGY

In this section we will describe a ‘Co-operative Market’ strategy designed to control allocation of SCP processing capacity in the network scenario described above. This strategy uses ‘Market-Based Control,’ a paradigm for distributed resource allocation based on application of concepts from economic science. Market-Based Control has been successfully applied to problems such as ATM bandwidth allocation [8] and power load management [9]; an overview of this expanding research area can be found in [10].

In our strategy, load control is carried out by means of tokens, which are ‘sold’ by Quantifier agents of ‘providers’ (SCPs) and ‘bought’ by Allocator agents of ‘customers’ (SSPs). The amount of tokens sold by an SCP controls the load offered to it, and the amount of tokens bought by an SSP determines how many IN service requests it can accept. ‘Trading’ of tokens (in an ‘auction’) is carried out such that the common good is maximised, hence the term *co-operative market*.

To describe the algorithm in detail the following notation is introduced. Let i , j , and k denote an arbitrary SCP, service class, and SSP respectively and let $r(j)$ denote the profit generated by servicing a class j request. All SSPs maintain IJ pools of tokens, one for each SCP and service class pairing. Each time SCP i is fed with a class j request one token is removed from the associated pool. An empty pool indicates that the associated SSP cannot accept more requests of the associated class. Pools are refilled at auctions, which take place every T time units (typically, T would be on the order of tens of seconds). Auctions are

centrally implemented by the Distributor using ‘bids’ (received in the form of messages every T time units) from all the Allocators and Quantifiers in the network.

Quantifier bids consist of the unclaimed processing capability over the coming period of T time units and the processing requirements for each service class respectively. Let $c_i, p_i(I), \dots, p_i(J)$ denote the bids from Quantifier i . Allocator bids consist of the number of IN service requests expected over the coming period of T time units for each service class respectively. Bids from Allocator k are denoted by $q_k(I), \dots, q_k(J)$.

The auction maximises the expected overall utility, measured as total profit, over the next T time units by maximising the expected marginal utility, measured as gain over cost, for every token issued. During the auction, a record is kept of the amount of capacity (in terms of tokens issued) sold on behalf of each Quantifier – let m_i be the processing capacity sold on behalf of SCP i . Similarly, a record is kept of the satisfied demands (in terms of tokens granted) of each Allocator – let $n_k(j)$ denote the number of tokens granted to SSP k for class j services.

The expected marginal gain $u_k(j)$ associated with allocating an additional class j token to Allocator k is defined as the profit associated with consuming it times the probability that it will be consumed over the auction interval. The latter may be obtained by interpreting the expected number of service requests $q_k(j)$ as the average of a Poisson distribution and we obtain

$$u_k(j) = r(j) \sum_{w=n_k(j)+1}^{\infty} q_k(j)^w / w! e^{-q_k(j)}$$

The expected marginal cost $v_i(j)$ associated with Quantifier i issuing an additional class j token is expressed as the utility it represents to the Allocators. We get a measure of this by summing over all services and Allocators:

$$v_i(j) = \sum_{j'=1}^J \sum_{k=1}^K \frac{p_i(j')}{p_i(j')} u_{k'}(j')$$

Let an (i,j,k) -allocation refer to assigning a token from SCP i with respect to service class j to SSP k . The expected marginal utility $w_{i,k}(j)$ of such an action is the gain to cost ratio:

$$w_{i,k}(j) = u_k(j) / v_i(j)$$

We note that a high marginal utility indicates that a large gain is obtained at a small cost. The triple i, j , and k which maximises the marginal utility therefore indicates the allocation which generates a maximum return on the spending.

The auction algorithm is initiated by setting $m_i = c_i$ for all i , $n_k(j) = 0$ for all j and k after which initial marginal gains $u_k(j)$ and costs $v_i(j)$ are computed for all for all i, j , and k . During the actual auctions tokens are handed out one by one from SCP i for service class j to SSP k , where i, j , and k are chosen to maximise the current values of $w_{i,k}(j)$. After each handout the remaining resources m_i , the remaining demands $n_k(j)$, and the marginal gains $u_k(j)$ and costs $v_i(j)$ are updated. Resources m_i are updated by subtracting the value of the issued token $p_i(j)$, demands $n_k(j)$ by adding one token, and gains and costs as shown above. The auction is completed when no more resources remain, i.e. when $m_i = 0$ for all i . At this point the Distributor will return the results to the Quantifiers and Allocators respectively. Quantifiers receive a JK -dimensional matrix of token allocations, and Allocators receive an IJ -dimensional

matrix of token allocations. It should be noted that token allocations are not additive between auctions – new allocations invalidate old ones.

Quantifier bids can be given in any appropriate unit, for example in our study the supply c_i is the number of microseconds the associated SCP is willing to spend on new service requests over the coming T time units, while the prices $p_i(j)$ are the number of microseconds associated with processing a request for a class j service. As long as the unit for supply and prices is the same there is no restriction on how bids are formulated (only relative values of supply and price are significant from the point of view of the auction process). Quantifiers know the processing capability of their SCPs and the processing requirements of the software corresponding to various service classes, and in its simplest form a bid is merely these values. More advanced forms where the capability is computed from filtered measurements and estimations of actually claimed resources are possible.

Allocator bids must all be given in the same unit, for example in our study they are the number of requests for a class j service expected over the coming T time units. Allocators count incoming requests, and in its simplest form a bid is merely the corresponding count for the T most recent time units. More advanced forms with filtering and prediction are possible. Since spending all tokens immediately during high loads would cause excessive delays, percentage thinning is applied such that the probability of accepting a class j request and thus spending a token is never higher than the number of remaining tokens over the number of requests expected during the remainder of the interval. There are no restrictions on how frequently this probability is updated.

Because of different clock rates, variable processing delays, and changing transmission delays, bidding, auctioning, and returning results cannot be exactly synchronised throughout an IN. However, even without perfect synchronicity, bids will be submitted, auctions held, and results returned approximately every T time units. Bids that arrive prior to or after an auction will be delayed until the next auction before results are returned, and this event can be used to regain maximum possible synchronisation. It is observed that the maximum delay that can be introduced by inaccurate timing is on the order of $T/2$. It is also noted that during overload, lack of synchronisation may contribute to a smoother flow of requests to SCPs if not all SSPs have their pools refilled at the same time.

7. PERFORMANCE EVALUATION OF PROPOSED STRATEGY

This section presents an evaluation of the likely performance of the Co-Operative Market (COM) strategy, based on the results of a simulation study. A previously studied Non-Agent Based (NAB) strategy, described in [3], has been simulated in parallel to the COM strategy in order to provide a benchmark for comparison.

7.1 Simulation Model Attributes

Our simulations modelled an IN configuration based on Figure 1, with $I = 8$ SCPs, $K = 32$ SSPs and a signalling network “cloud” with characteristics as described in §5. Both SCPs and SSPs are assumed to be statistically identical. The processors at the SCPs are engineered to operate at a nominal load, r_{nom} of 35% and with a maximum permissible load r_{max} of 90%. The

Distributor runs auctions every $T = 10$ s using the algorithm and procedures described in §6. Percentage thinning probabilities are updated every $T/10 = 1$ s.

The network supports $J = 2$ service classes the signalling procedures and processing requirements of which are modified versions of the two services defined in [11]. Successful sessions of service 2 are assumed to be 10 times more profitable than those of service 1. The call holding times of both services are negative exponentially distributed about a mean of 100s. Calls consist of a connection and a disconnection phase. The processing load inflicted by the connections is for service 1 4.1ms and for service 2 6.4ms and the disconnection load is 1.4ms and 0.16ms for services 1 and 2 respectively. Service requests arrive according to independent Poisson processes with a rate $I_{j,k}$ requests per second for each service class j and SSP k respectively.

7.2 Simulation Results and Evaluation

The criteria we use for evaluation of the mechanism are its ability to satisfy the five objectives below. These objectives encompass three separate viewpoints of network performance. These are the Operator viewpoint, the User viewpoint and the network itself.

Objective 1. SCP load levels remain as close to the target load of $\hat{r} = 0.9$ Erlangs without ever exceeding it;

Objective 2. Network profit should be maximised;

Objective 3. Response delays for network users should be minimised;

Objective 4. The mechanism should be fair in its treatment both of services and users – accepted service sessions should experience similar response delays, regardless of service type or location of the user in the network;

Objective 5. Load should be balanced equally between all SCPs in the network.

All graphs we will present are based on average values from 5 independent simulation runs. Confidence intervals are of the order of $\pm 2\%$, for clarity these are not represented in the graphs.

The graphs presented in Figures 2 to 6 below correspond to the results of a number of simulation runs in which the arrival rates $I_{j,k}$ were varied so that offered load to each SCP varied in the region of 0 (no traffic) to 2.0 Erlangs (severe overload).

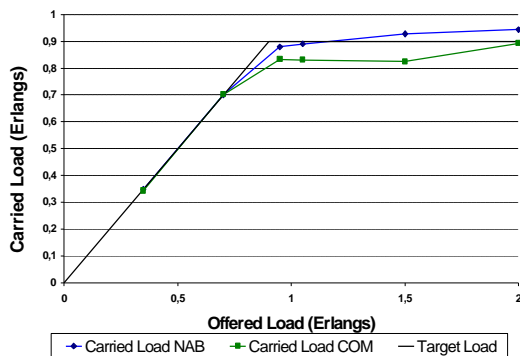


Figure 2: Carried Load

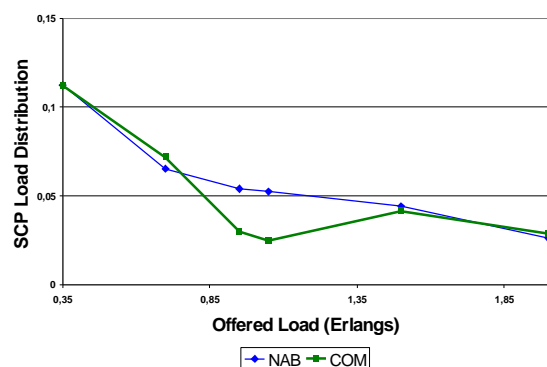


Figure 3: Relative SCP Load

Figure 2 shows the mean carried SCP load, averaged over all SCPs in the network and the ‘ideal’ target load for a range of offered loads for both strategies. It can be seen that in normal load conditions the operation of the strategy has negligible effect on SCP carried loads. The inability to reach the target load at offered loads exceeding 0.75 Erlangs is due to a somewhat non-optimal distribution of tokens among the SSPs. The figure shows that the COM strategy partly fulfils *Objective 1* in that on average the target SCP load level is not exceeded, however there is some room for improvement in terms of keeping the carried load closer to the desired level during overload.

Figure 3 shows the largest average load difference between the SCPs during a simulation. Individual differences are normalised against the average load of all SCPs during the simulation. The load span decreases with offered load simply due the reduced probability of an SCP being idle. Even at low offered loads, 0.35 Erlangs, is the load span quite small, with average load varying by $\pm 5.5\%$. Thus *Objective 5* is satisfied by both strategies.

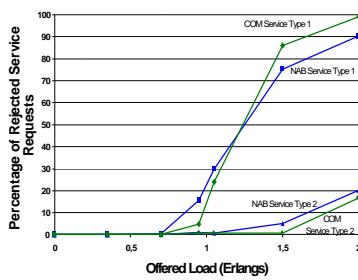


Figure 4: Call Rejection Probability

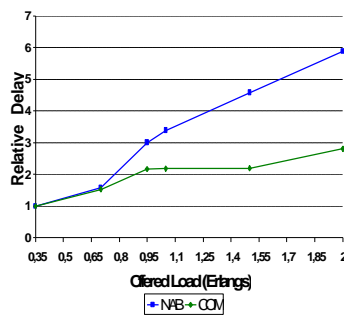


Figure 5: Relative Connection delays

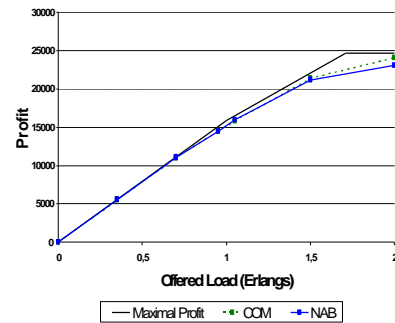


Figure 6: Network Profit

Figure 4 shows the percentage of rejected requests for both service types. Once again it can be seen that in normal load conditions the strategies have no effect – no service requests are rejected. Once throttling commences, only requests for service 1 (the low profit service type) are rejected. This means, that through its throttling policy, the COM strategy seeks to maximise profit and thus goes a significant way towards meeting *Objective 2*. The COM strategy is clearly more aggressive in meeting the profit and load objectives than the NAB strategy. The latter can be seen in the average rejection probability at 2.0 Erlangs where the COM rejects 55% and the NAB 52% of all arrivals.

Mean relative values of the response times to the initial request, connection delay, of a service session are shown in Figure 5. This graph shows that even in periods of extreme overload the connection delay experienced by a user is only approximately 3 times that experienced in periods of normal load for the COM strategy. Given the relationship between typical telephony service connection delays customer expectations [12], this appears to be an acceptable value, so the strategy also meets *Objective 3*. In addition, results not presented here show that *Objective 4* is also met – the difference in relative connection delays for accepted sessions of both types is minimal, thus all sessions are treated in a fair manner. The considerably longer connection delays for the NAB strategy is attributed to a higher probability of an SCP experiencing the load of 1.0 Erlang for a longer period of time.

The network profit is presented in Figure 6. The COM strategy marginally out-performs the NAB strategy in maximising profit. The profit follows the optimal curve up to a load of 1.5 Erlangs, thus *Objective 2* is satisfied. It is noted that the maximum profit is achieved at the load where the demand for service 2 is sufficient to saturate all SCPs.

The graphs presented in Figures 7 to 11 below correspond to the results of a number of simulation runs in which the arrival rates $I_{j,k}$ were varied from an offered load of 0.35 Erlangs to 0.9 or 2.0 Erlangs in a step. The offered load was reduced to 0.35 Erlangs after 300s.

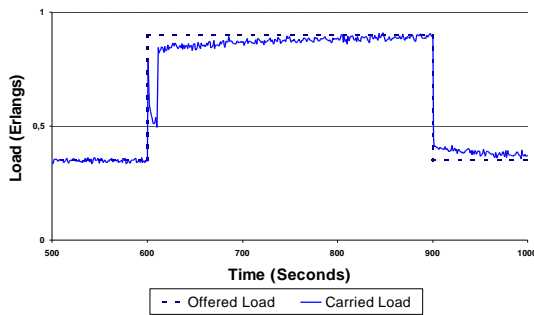


Figure 7: COM, A step to 0.9 Erlangs

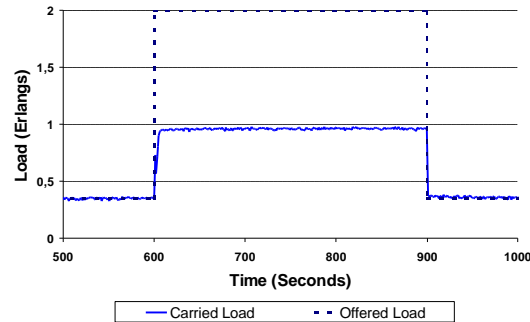


Figure 8: NAB, A step to 2.0

Figure 7 shows the response of the network when the COM strategy engaged to a step increase in offered load to a peak load of 0.9 Erlangs. The two services contribute equally to the load. The COM strategy is unable to cope with the load step during the auction interval immediately following the load step. All SCP capacity has been allocated to the SSPs in the auction, but the Allocators that submitted the largest bids in terms of capacity demands have been granted excess tokens compared to these bids and, thus, the network becomes unable to perform optimally as some SSPs can not spend all allocated capacity while others have too little. The slow build-up in carried load, from 0.83 Erlangs to 0.9 Erlangs, is the effect over time of the 100 s mean call holding times (the average capacity requirement for a disconnection make up around 15% of the total and is included in the bid for capacity in the auctions.) A similar explanation is valid for the slow reduction of carried load after the step, from 0.4 Erlangs to 0.35 Erlangs. The COM strategy is eventually able to master average carried load at 0.9 Erlangs from the second auction and onwards.

Figure 8 shows response of the network for a step of offered load from 0.35 to 2.0 Erlangs when the NAB strategy is employed. The load is evenly distributed between the two services. It does not experience the dip as seen in the previous case after the transient but it is a bit slow in reacting to the increase of load. The NAB strategy is also less accurate in maintaining the carried load at 0.9 Erlangs. The effects of the 100 s mean call holding times are also visible in the NAB case. The networks response is in all important aspects unaltered when one service alone is responsible for the increase of load.

Figure 9 below shows the COM strategy with a step increase of offered load for service 1 only, from 0.35 to 2.0 Erlangs. The response is very similar to that in Figure 7. The dip of the first auction after the step is there and so are the effects of disconnections. The average carried load falls short of the desired 0.9 Erlangs even after the initial disconnection effects. The dip is slightly deeper as token allocations are equally split between service 1 and 2 but SSPs can only

make use of tokens for service 1, which only require 80% of the capacity required for the average of both services.

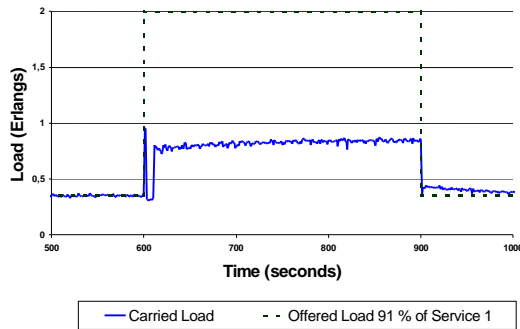


Figure 9: COM, A step to 2.0 Erlangs

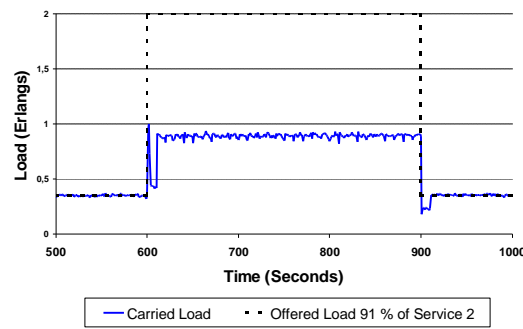


Figure 10: COM, A step to 2.0 Erlangs

Figure 10 shows the COM strategy with a step increase of offered load, only service 2, from 0.35 to 2.0 Erlangs. The response displays similarities to that in figure 7. The dip of the first auction after the step is there, not so deep due to a 20% higher requirement on processing capacity compared to an average service. The network is unable to spend the tokens allocated for service 1 during the dip. The effect of disconnections is negligible, as disconnection load for service 2 is only makes up for 2.5% of the total. The average load is maintained at a 0.9 Erlangs, fulfilling both *Objectives 1* and *2*. The dip in carried load after the negative edge is explained by a certain greediness of the strategy – mostly service 2 tokens are distributed as the strategy is attempting to maximise profit (*Objective 2*) by satisfying the expected huge demand for service 2.

8 CONCLUSIONS AND FUTURE WORK

In this paper we have presented some arguments as to why Agent Technology may provide a suitable methodology for developing load control strategies that can achieve, in an efficient and flexible manner, the balancing of traffic load and avoidance of harmful overloads in complex, dynamic network environments. As an example of the application of the agent-based approach, we presented an IN load control strategy that was based on market-based control, a paradigm which has been successfully applied to a number of application fields and which is particularly suited to implementation as a community of communicating agents.

We also presented the results of a simulation study which demonstrated that the proposed strategy performs well, but not flawlessly, under a variety of load situations – in terms of keeping load at acceptable levels, maximising network profit, maintaining acceptable response times and treating service types in a fair manner. Future work will concentrate on enhancing this strategy in order to provide a more convincing demonstration of the power of the agent-based approach. Of paramount importance is extension to decentralise the auction process into a number of independent auctions.

In addition, the strategy will be modified to allow for adaptation to changes in the network environment such as the introduction/withdrawal of services or resources (SCPs). It will also be modified to facilitate its application in multi-operator domains, for example a Distributor

could participate on behalf of Allocators/Quantifiers in an auction run by the Distributor of another domain. In doing so it could perform negotiations on common semantics (for example Service type 1 in Network A = Service Type 7 in Domain B), which would be made much simpler because of the use of a speech-act based Agent Communication Language.

ACKNOWLEDGEMENTS

Parts of this work were supported by the EC-funded ACTS project MARINER [13]; the authors wish to acknowledge the valuable contribution of their colleagues in this project.

REFERENCES

- [1] N. Mitra & R. Brennan, "*Design of the CORBA/TC Inter-working Gateway*," to appear in Proc. IS&N'99, Barcelona, April 1999.
- [2] F. Lodge, D. Botvich, T. Curran, "*Using Revenue Optimisation for the Maximisation of Intelligent Network Performance*," appears elsewhere in this volume.
- [3] Å. Arvidsson, L. Angelin and S. Pettersson, "*Congestion Control in Intelligent Networks for Real Time Performance and Profit Optimisation*," Proc. ITC 15, Edited by V. Ramaswami and P.E. Wirth, Elsevier, 1997.
- [4] A.H. Atai, B.S. Northcote, "*AIN Focused Overloads - A review of USA CCS network failures and lessons learned*," Proc. ITC Mini-Seminar on Engineering and Congestion Control in Intelligent Networks, Melbourne, April 1996.
- [5] B. Jennings, F. Lodge, T. Curran, "*A Strategy For The Resolution of Intelligent Network (IN) and Signalling System No. 7 (SS7) Congestion Control Conflicts*," Proc. ICC'98, Atlanta, USA, June 1998.
- [6] M. Wooldridge & N. Jennings, "*Intelligent Agents: Theory and Practice*," The Knowledge Engineering Review Vol. 10, No. 2, 1995, pp. 115-152.
- [7] R. Brennan, B. Jennings, T. Curran, "*SS.7 as an Agent Platform Message Transport Protocol*," to appear in Proc. IS&N'99, Barcelona, April 1999.
- [8] M. Gibney, N. Jennings, "*Market Based Multi-Agent Systems for ATM network Management*," Proc. 4th Communications Network Symposium, Manchester, 1997.
- [9] F. Ygge, "*Market-Oriented Programming and its Application to Power Load Management*," University of Lund, Sweden, Ph.D. Thesis, 1998.
- [10] S. H. Clearwater (Ed.), "*Market-Based Control*," World Scientific, 1996.
- [11] G. Karagiannis, V. F. Nicola, and I. G. M. M. Niemegeers, "*Quantitative Evaluation of Scalability in Broadband Intelligent Networks*," Proc. Performance of Information and Communication Systems, Körner & Nilsson eds., Chapman & Hall, London 1998, pp. 65-82.
- [12] D. MacDonald, S. Archambault, "*Using Customer Expectation in Planning the Intelligent Network*," Proc. ITC 14, Juan les Pins, 1994.
- [13] ACTS Project MARINER, information available at <http://www.teltec.dcu.ie/mariner>.