An Evaluation Mechanism for QoS Management in Wireless Systems

Behzad Bordbar[1], Rachid Anane[2] and Kozo Okano[3]

[1]School of Computer Science, University of Birmingham, UK.
B.Bordbar@cs.bham.ac.uk
[2]School of Mathematical and Information Sciences, Coventry University, UK.
R.Anane@coventry.ac.uk
[3]Graduate School of Information Science and Technology, Osaka University
Okano@ist.osaka-u.ac.jp

Abstract

The evaluation of QoS requirements is one of the critical functions that span both the design and the run-time phases of QoS management. This paper presents an architecture for QoS evaluation and admission control, based on the modelling of both system behaviour and QoS requirements. Two aspects are considered. The first refers to QoS management, and to the component-based architecture for QoS evaluation. The second relates to the approach and its illustration by a case study, based on a Personal Area Network. The proposed approach relies on the instantiation of models for representing both the behaviour and the QoS aspects of the system in terms of Timed Automata. The compatibility of the evaluation mechanism with architectures with a defined role for a QoS manager, such as ITSUMO, is also highlighted.

1. Introduction

The flexibility afforded by IP mobility in distributed systems is often at odds with the challenge of ensuring continuity of service and maintaining an agreed level of QoS [6]. The highly dynamic nature of distributed systems can be mediated by a negotiation phase between clients and QoS managers to reflect prevailing conditions. The outcome can be expressed in terms of a service level agreement (SLA), which is often translated into a service level specification (SLS), and from which the QoS parameters are extracted. In addition to the inherent channel errors, user mobility and contention between users for scarce resources may lead to situations where a service level may not be honoured by nodes, when handover takes place [7]. Handover is symptomatic of the complexity of QoS management because of its implications for QoS provision. It may lead to the re-negotiation of service levels and to the re-allocation of resources, a disruption that may increase network latency [13]. This characteristic has resulted in an increasing interest in QoS management architectures, which allow accurate and efficient evaluation of QoS requests, based on global QoS information. This paper is concerned with the presentation of an architecture for QoS evaluation.

The remainder of the paper is organised as follows. Section 2 gives an introduction to QoS management. Section 3 describes the architecture of a QoS evaluation mechanism. Section 4 illustrates the proposed approach with a case study. Section 5 discusses issues related to the architecture, and Section 6 concludes the paper.

2. QoS management and architecture

Support for seamless mobility and adaptive computing in QoS provision are important requirements of QoS management. In particular, the transition period generated by a handover needs to be managed by the transfer of the SLS of mobile stations between adjacent nodes. Transfer can follow a reactive approach and be performed on demand such as in the architecture proposed in [18]. ITSUMO [8][10], on the other hand, promotes a proactive approach; an SLS, once determined, is broadcast to all nodes in the same domain in order to ensure a seamless handover. In many QoS architectures tables are focal points of activity especially in admission control, or when re-negotiation is mandated [5]. Sugawara *et al* [17] present an example of a table-based implementation, where information about QoS levels is maintained. The QoS table holds the resources required by all the scheduled tasks in the system, and its purpose is to facilitate the resource allocation to tasks and the determination of system resource requirement.

The Global Quality Server (GQS) in ITSUMO keeps various items of information including SLAs, patterns of mobility and resource availability. The table is used by the GQS, as QoS manager, for negotiating with mobile stations their requirements. In addition to this centralised information, each QoS Local Node (QLN) holds a subset of information in a local table, which is updated frequently by the GQS. Similarly, in the architecture proposed by Pau *et al* [15], the Wireless Quality Enhancer (WQE), a QoS manager and policy maker makes use of a table for

interacting with the access points (AP), which are policy implementers. The tables are also directly relevant to the modelling and validation of QoS requests. One advantage of tables in QoS schemes is that policy-based QoS management can be enforced [14]. The approach presented in this paper extends the table-based approach by taking into account system behaviour.

3. A QoS evaluation mechanism

The scope of the proposed architecture is determined by the desire to enhance admission control in QoS management.

3.1 Architectural components
The proposed architecture is presented in Figure 1. On receipt of a QoS request the QoS Manager calls upon the QoS Evaluation Module (QEM) to determine whether a QoS request can be satisfied. The evaluation process involves a number of components as follows.

Table of Commitments (TOC): holds information on the current state of the system. This includes information about system nodes, resources and QoS allocated to them. TOC can be used to create a model representing the behaviour of the system and a model representing the QoS. The information in TOC is logged in a repository for optimisation purposes.

Repository: holds previously instantiated models and their requested QoS level. This allows the system to look up a QoS when a previous situation arises again. In the case of a new scenario the evaluation is delegated to a component called QEM.

QoS Evaluation Module (QEM): The evaluation process requires three elements. First, a model of the behaviour of the system, which includes the behaviour of the existing system components, and also the behaviour of the new user. Second, to verify a QoS request, a formal representation of such expression is generated. Third, the QoS requirements must be checked against the behavioural model. This is achieved by the use of QoS Resolution Engine, explained below.

Behavioural Model Repository (BMR): BMR is a repository that contains various templates, which represent the models of behaviour of components such as communication protocols and channels. The templates are the building blocks from which the overall behaviour of the system can be composed. QEM uses the templates in BMR to instantiate different parts of a model, and creates a behavioural model for the overall system.

QoS Model Repository (QMR): Similarly, QMR consist of a set of templates that can be used to provide models for a QoS statement. QEM uses the templates in the QMR to instantiate formal representations of QoS aspects of the system.

QoS Resolution Engine (QRE): QRE operates on the behavioural and QoS models generated from BMR and QMR and TOC. A QRE is the intelligent component that receives a model of the behaviour and a model of the QoS request and checks the validity of the QoS statement against the behaviour of the model. The QoS request may be an aggregation of the QoS request stored in the table of commitments.
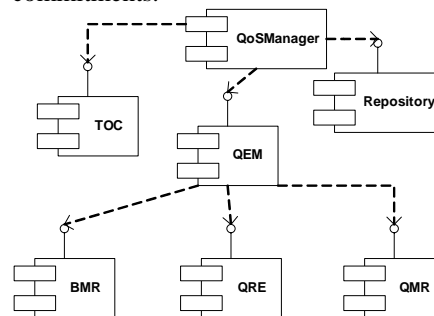

Figure 1: An architecture for QoS evaluation

3.2 Implementation of architecture
In an earlier paper [3], an implementation of the architecture, with the focus on the BMR, QMR and QRE, was outlined in terms of Timed Automata. BMR includes various templates Timed Automata [12] for source, sink, different types of buffer, decoder [4] as well as communication protocol [2]. These are the building blocks for the creation of the behavioural models. A template Timed Automata represent the behaviour of the sub-components and include parameters for the variables of the model. The behavioural models of the system are networks of Timed Automata [12], aggregated from the instantiation of templates in BMR, by assigning values to parameters in each template. QMR, on the other hand, is a repository of template Timed Automata corresponding to various Timeliness QoS properties such as jitter, latency and throughput [4][8]. These are to be used as Test Timed Automata [1]. A Test Timed Automaton instantiated from the templates can be used to verify the corresponding QoS statement against the behaviour of the system, as modelled by instantiations from the BMR [3]. The final component, QRE is based on the model checker UPPAAL [12][19], which can perform the verification of the model. To ensure compatibility with the UPPAAL files, which are stored as XML files, the concrete representation of the Timed Automata is in XML. Despite this bias towards XML as a specific representation, the main implication is that XML is also a suitable form for holding

information in the Repository and the Table of Commitment (TOC).

4. Case Study: a Personal Area Network

This section is a case study to illustrate the behavioural and the QoS modelling process, and the evaluation mechanism.

4.1 Scenario

Let us consider a Personal Area Network (PAN), which consists of a Wireless router connected to the Internet. Figure 2 depicts a number of users (Stations), namely two PCs (PC1, PC2) and a Laptop (L1), which access the Internet via a router (Access Point). L2 is not part of the initial configuration. The stations are competing with each other to acquire bandwidth and to achieve a better QoS.

Now, consider a laptop L2, which is joining the PAN as depicted in Figure 2. The task of the QoS manager is to determine the effect of a provision of the service to L2 on the existing components. From the information it holds and the description of the behaviour of L2 and its QoS, the QoS Manager can create a model of the overall system. Let us refer to the model that includes the description of the behaviour of L1, L2, PC1 and PC2 as *m*. Under the behaviour specified in *m*, the system must not only satisfy the new QoS request from L2, but also the committed QoS requests for L1, PC1 and PC2. Such QoS requirements (including the request from L2) will be referred to as *q*.
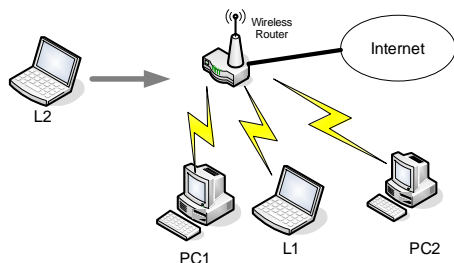

Figure 2: Wireless Router used in a PAN

Given the dynamic nature of wireless networks, it is possible that the above scenario, modelled as *m* and *q*, may have occurred before. In this case, the QoS Manager may find such information in the Repository. It can retrieve the models and use them to decide if QoS *q* is achievable within the behavioural model *m*. If, on the other hand, the models are not in the Repository, the evaluation process goes through a number of steps, detailed as follows. QEM receives a request to check if *q* is valid. The request includes the parameters representing the QoS statement and a model of the behaviour of the system, which can be generated

from the information in TOC. QEM then instructs BMR and QMR to instantiate the behavioural model and the QoS statement, which are transferred to the QRE to *check* and returns the *result* to QEM.

4.2 Components and Behaviour

The modelling process requires an explicit identification of the components of the network, and their interactions. To this end, it was decided to focus on the case where the applications on the station are *just* downloading packets from the Internet, i.e. there is negligible or no traffic from any station towards the router. As depicted in Figure 2, the Internet is the provider of the *packets*. The Wireless Router sends the packets to the Stations. Each station is assumed to contain an Input Module, which receives the packets from the Wireless Router and passes it to the Application Layer. The Application Layer represents a group of applications, which are viewed as "consumers of packets". For the sake of clarity caching and various other protocols involved in the transfer of the packets were not included. In order to communicate with the stations the Wireless Router needs to access the medium by means of a protocol. The wireless local area network (802.11) [11] defines three basic access mechanisms. This example adopts the Point Coordinate Function (PCF) as the access mechanism. For further information on the WLAN and PCF, the reader is referred to [16].

4.3 Modelling behaviour

The modelling of a system's behaviour is an aggregation of the behavioural models of its components. This section presents a brief description of the behavioural models of the components in terms of networks of Timed Automata [12] as depicted in Figure 3. Due to space restrictions, only the Timed Automata "*TA for PCF*" will be explained.

TA for PCF: The interaction with the router, which makes use of PCF, is modelled as *TA for PCF*. At the start of a contention free period, the medium gets busy, and this is shown with the signal *access?* of *TA for PCF*. The integer value *i* ranges over the number of stations. There are N stations, i.e. $i = 1, ... , N$. Depending on the value of *i*, the downlink (*data!*) is meant to be delivered to station number *i*. The start with value of *i* is 1 and, it is incremented each time before the data is delivered to the next station.

After gaining access to the medium, the PCF sends data to the station. The data sent by the DCF must be broken into units of maximum length of MAC Service Data Unit (MDSU) [11][16]. *A* denotes the amount of time required for the MDSU to reach the destination. As a result, at state *Sending_Data*, within *A* unit of time *data!* is sent. Depending on the value of *i*, the signal *data?* is used in the Application Layer

of Station i. When the transmission of data finishes, an urgent acting *CF-poll* signal is sent to mark the end of data. To notify the medium, an *idle!* signal is sent to mark the end of access. Then the PCF waits for SIFS (SIFS is 10 ms[1]). At exactly SIFS units it receives a *CF_ACK!* signal from the Station that the data has been received. However, if $i < N$, in order to ensure that the next downstream goes to station $i+1$, the value of i is incremented. If $i = N$, this indicates that one contention free period is finished and a *CF-end* signal is sent. In this case, since no contention period is used, the CF-end is replaced with a simple acknowledgement signal *CF_ACK*. If the *CF_ACK* is sent a back-off period of SIFS is required.
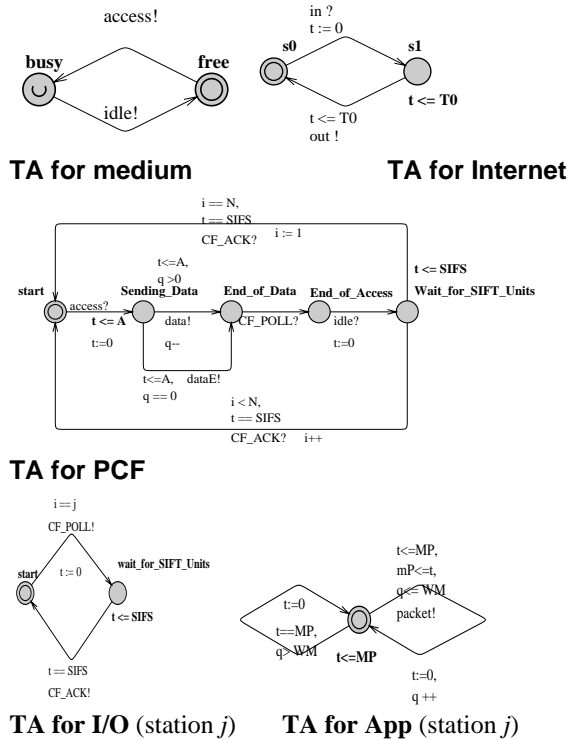

TA for medium   **TA for Internet**

TA for PCF

TA for I/O (station j)   **TA for App** (station j)

Figure 3: Behavioural Model of the system

4.4 QoS modelling and Verification

Consider the system of the previous section, which consists of two PCs and a laptop, and with the parameters, WM = 5, WM = 5, SIFS = 40, PA = 20, BD = 20 and Ad (application delay) of 5. Suppose one of the requirements is that the throughput (Anchored throughput [4]) is at least 1 frame every 127 units for laptop 1 (LP1), i.e. the occurrence of at least 6 signal VSChunk! in each period of SIFS*19, see Figure 3 *"TA for I/O"*.

Now, assume that a new laptop LP2, with a specification identical to LP1 wants to join the

system. Also, assume that LP2 requests the same level of throughput (at least 1 frame per 127 unit). If the scenario involving LP1, LP2, PC1 and PC2 has not been modelled before, i.e. is not held in the TOC, the QoS Manager has to evaluate the achievability of the QoS for LP2 using the QEM. In order to do so a model of the system is created from the templates in BMR. This includes using the templates depicted in Figure 3 and the numerical parameters to create a network of Timed Automata model of the system. In order to check the QoS required by LP2, a Test Timed Automata for anchored throughput is required, as depicted in Figure 4. For further details on Test Timed Automata for the verification of QoS we refer the reader to [4]. For this case, it can be seen that anchored throughput of at least 1 frame per 127 unit is not achievable.
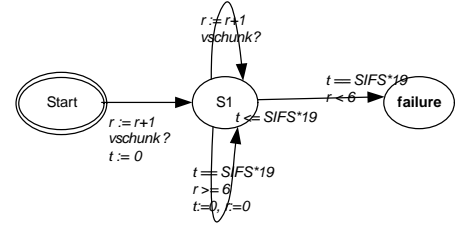

Figure 4: TA for Anchored Throughput

It is possible for LP2 to negotiate with the QoS manager and request a lower level of QoS. For example, it can be checked that the system can provide at least 1 frame per 167 unit (at least 6 signal VSChunk! in each period of SIFS*25).
We have conducted various tests on non-anchored throughput and jitter, in UPPAAL 3.4 running on an Intel III 600MHz Linux machine.

5. Discussion and further work

The proposed architecture was motivated by two major concerns, namely the need to hold QoS information about commitments and requests, and the ability to determine accurately the viability of new QoS requests. The first goal led to the introduction of a table of commitments for current requests, and a repository for previous ones. The evaluation and viability issue was addressed, firstly by instantiating existing templates into models of behaviour and QoS, and secondly by verifying the viability of these models. This approach simplifies the interactions between client applications and the centralised QoS server. Since the server is in possession of global information and is aware of behavioural models, QoS negotiation is minimised. A proactive approach can also be fostered by the existence of the repository. Concomitant with a transfer of complexity from application to server is a better assessment of

---

[1] It is 10 ms if FHSS is used and 28 ms if DSSS is used.

resource management and allocation. Furthermore, the centralised approach allows for the enhancement of the level of intelligence in QoS management; a prospect that is in tune with the aims of the GQS in ITSUMO. Although a centralised decision maker can be a potential bottleneck, its role is mandated by user mobility and by traffic level.

This affinity with ITSUMO is also underlined by a clear separation between QoS decisions and resource allocation. It is also the modelling approach that confers to the proposed architecture the capacity for providing support for flexibility, scalability and adaptivity in dynamic QoS management. This feature was illustrated by the case study. Moreover, the autonomy enjoyed by the QoS manager owes much to the availability of templates, the instantiation of models and their verification. These features may, however, be available at a price. There is a need to keep the template repositories up to date, and to optimise their symbiotic relationship with the table of commitments.

The concerns outlined above are pointers for further work. Of particular significance is the investigation of suitable structures for the table of commitments and for the repository, and the identification and selection of methods and adequate techniques for representing and manipulating models. The ability to aggregate and disaggregate instantiated models is a requirement for an efficient implementation of re-negotiation. Future work will also involve the investigation and evaluation of other formalisms for modelling behaviour.

6. Conclusion

This work has highlighted the importance of modelling and verification in the accurate evaluation of QoS requests in wireless systems. It has shown that this can be achieved by a symbiotic relationship between table manipulation and model building. It has also pointed out that efficiency requirements for wireless systems, in highly dynamic environments, warrant a careful investigation and selection of both formalisms and mechanisms in QoS management.

References

[1] L. Ageto, P. Bouyer, A. Burgueo and K. G. Larsen The Power of Reachability Testing for Timed Automata, *in Theoretical Computer Science 300(1-3) 411-475*, 2003.

[2] J Bengtsson, W. O. D. Griffioen, K. J. Kristoffersen, K. G. Larsen, F. Larsson, P. Pettersson, and Wang Yi Automated Analysis of an Audio Control Protocol Using Uppaal, J *In Journal of Logic and Algebraic Programming, volumes 52-53*, pages 163-181, Holger Hermanns and Joost-Pieter Katoen (eds.), 2002

[3] B.Bordbar and R. Anane. An Architecture for Automated QoS Resolution in Wireless Systems, *Proceedings of the IEEE AINA 2005 International Workshop on Web and Mobile Information Systems (WAMIS),* March 2005, Taipei, Taiwan, 774-779.

[4] B. Bordbar and K. Okano. Verification of Timeliness QoS Properties in Multimedia Systems, *Proceeding of 5th International Conference on Formal Engineering Methods, Lecture notes in Computer Science*, pp 523-540, 2003

[5] R. S. Cardoso and F. Kon, A mobile Agent Infrastructure for QoS Negotiation of Adaptive Distributed Applications, *CoopIS/DOA/ODBASE (2)* 2004, 1590-1607

[6] Burness, L., Hepworth, E., Lopez, A. and Manner, J., Architecture for Providing QoS in an IP-based Mobile Network*, Proc. IST Mobile Communications Summit* 2001, 9 -12 September, Barcelona, Spain

[7] C.D Cavanaugh, L.R. Welch, B.A. Shirazi, E. Huh and S. Anwar, Quality of Service Negotiation for Distributed, Dynamic Real-time Systems*, Parallel and Distributed Processing 15 IDPDS 2000 Workshops, LNCS 1800*, Cancun Mexico, 2000, 757-765.

[8] D. Chalmers, M. Sloman: A Survey of Quality of Service in Mobile Computing Environments, *IEEE Communications Surveys*, 2nd Quarter 1999,

[9] J.C. Chen, A. McAuley, A. Caro, S. Baba, Y. Ohba, P. Ramanathan. A QoS Architecture for Future Wireless IP Networks. *In Twelfth IASTED International Conference onParallel and Distributed Computing and Systems (PDCS 2000),* Las Vegas, NV, 2000

[10] J. C. Chen, A. McAuley, V. Sarangan, S. Baba, and Y. Ohba, Dynamic Service Negotiation Protocol (DSNP) and Wireless DiffServ, *ICC'02, New York city,* 2002.

[11] IEEE Computer Society. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Standard 802.11*, 1999.

[12] K. G. Larsen, Paul Pettersson and W. Yi UPPAAL in a Nutshell *Springer International Journal of Software Tools for Technology, 1(1+2),* 1997

[13] S. Lu, K-W Lee and V. Bharghavan, Adaptive Service in Mobile Computing Environments, *IFIP International Workshop on Quality of Service*, 1997

[14] P. Nanda and A. Simmonds, Policy Based Architecture for QoS over Differentiated Services Network. *International Conference on Internet Computing,* Las Vegas, 2003, 866-872

[15] G. Pau, D. Maniezzo, S. Das, Y. Lim, J. Pyon, H. Yu, M. Gerla, A Cross-Layer Framework for Wireless LAN QoS Support, *IEEE International Conference on Information Technology Research and Education*, New Jersey, USA, August 10-13, 2003.

[16] J. H. Schiller, Mobile Communications, 2nd Edition, Addison-Wesley, 2003

[17] T. Sugawara and K. Tatsukawa, Table-based QoS Control for Embedded Real-Time Systems, *ACM Workshop on Languages, Compilers and Tools for Embedded Systems*, 1999

[18] G. Stattenberger and T. Braun, QoS Provisioning for Mobile IP Users: Applications and Services in Wireless Networks, *Hermes Science*, 2001

[19] UPPAAL www.uppaal.com