A General Model for QoS Adaptation

D. G. Waddington and D. Hutchison
Computing Department,
Lancaster University,
Lancaster LA1 4YR, UK
e-mail: [dan, dh]@comp.lancs .ac.uk

ABSTRACT
New and more advanced applications are supporting end-to-end Quality of Service (QoS) guarantees through the configuration and management of distributed resources. As an effect of the sharing of static resources across multiple concerns, coupled with the use of dynamically changing resources such as mobile communications, the general availability of resources in a distributed environment is variable and potentially unpredictable. In this paper we discuss the requirements for QoS adaptation mechanisms and QoS-based distributed resource management, together with our approaches to QoS monitoring and adaptation, in the context of our recently developed Distributed Resource Management Architecture (DRMA).

1. Introduction

The growth of general computing performance is being closely followed by the exploitation of this capability in more and more complex applications. Many of these applications, such as video conferencing and distributed collaborative environments, have dynamically changing and potentially unpredictable QoS requirements. This problem is exacerbated by the heterogeneous nature and varying capabilities of today's end-systems and network infrastructures. As a result conventional resource reservation and admission techniques cannot guarantee QoS without considerable over-booking and inefficient resource utilisation, something which is particularly undesirable in systems that are required to maintain high levels of resource sharing.

To address this problem, and avoid any adverse impact to the end-user, applications and their infrastructure need to become adaptive. This means that either the application must tolerate fluctuations in resource availability or that the supporting infrastructure can itself mould, through distributed QoS management, to the dynamically changing requirements of its applications. Furthermore, certain applications simply cannot operate outside strict resource requirements, and therefore the need for QoS management arises. 'QoS management' encompasses the maintenance of a required level of service through the co-ordinated configuration and control of end-to-end resources. Because of the obvious proliferation and acceptance of distributed object computing, and more importantly its usefulness in addressing the problem of QoS management in an open distributed system, proposals have already been made for object-based management frameworks [Dang, 95][ISO, 97]. However, up till now, much of the work has only addressed issues of QoS specification and the projection of useful QoS abstractions and services to the application programmer.

2. Providing QoS by Resource Management

In order to sustain the QoS requirements of a given continuous media application, all resources involved in the handling and processing of data from end-to-end, must be carefully co-ordinated and managed. End-to-end QoS is some function of the resource utilisation of a distributed application, from client through network to server.

Any application which needs to provide a QoS-bound service must maintain its distributed resource utilisation within a finite space, which we call the *resource capacity region*. Figure 1 illustrates a possible relationship between resource usage in the end-systems and the network. The region's surface represents the balance of resources required to sustain a particular end-to-end QoS; hence each level of service has associated a different capacity region. For each level of service, provided that an application maintains its resource utilisation within the defined region, QoS is sustained (a concept of capacity regions was proposed by Columbia University's work [Hyman, 95] on meeting end-to-end QoS guarantees over high performance packet-switched networks).

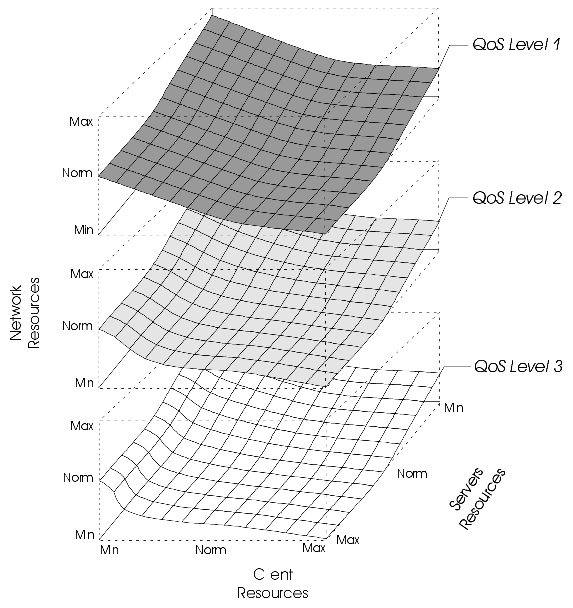

Figure 1. Resource Capacity Regions for QoS Levels

If an application is unable to maintain its utilisation within the region, possibly due to the unavailability of resources, then degradation in the application QoS will occur. On the other hand, if an application's resource utilisation is not close to the surface of the capacity region, then resources are not being used optimally, which may result in the degradation of QoS for other applications sharing the resources. Thus, distributed applications providing end-to-end QoS should strive to maintain resource utilisation as close as possible to the balance defined by the surface of the resource capacity region. So far, our discussion of the resource capacity regions has been limited to distributed applications based upon single client/network/server scenarios. However, the concept can be readily extended to multi-point communication scenarios, resulting in additional dimensions to the capacity graph (we are not limited by 3-dimensional space, as capacity regions are actually realised as non-visual multi-variable relationships).

QoS adaptation is the process of maintenance control, facilitated through alterations to either the balance and distribution of resources or to the application's level of service, on short time scales. Adaptation processes often occur as a result of *QoS notifications*, usually emitted from QoS monitoring mechanisms, which indicate a change in the observed service affected through the availability of some element of the end-to-end resources. Notifications may indicate a imminent lack of resources and hence reduction in service quality (QoS degradation) or a failure to maintain service quality through a complete loss of resources (QoS failure). Whether degradation or actual failure occurs, QoS adaptation is required to either adjust the balance of resources to maintain graceful degradation, recover service quality, or alternatively inform the end-user of the need to alter to a new level of service (change in level of service could entail dropping one or more media channels, or reducing information resolution). Ideally adjustments in resource balance, affected by the adaptation mechanisms, will satisfy the function of resource utilisation described by the surface of the resource capacity region. In doing so this is likely to involve one or more entities of the resource set (client, server or network) increasing their own resource utilisation to counteract deficiencies in the failing entity. It is also important to realise that the rules of adaptation that we apply to simple client-network-server applications can be readily scaled to applications which communicate in one-to-many and many-to-many relationships.

2.1 Hierarchical QoS Management

From an engineering perspective, QoS management in a distributed system is a substantially complex task. An approach which has been proposed in the Distributed Resource Management Architecture (DRMA) [Waddington, 97] is hierarchical QoS management. This technique breaks down the task of managing end-to-end resources by dividing the problem into a set of finer-grained point-to-point requirements which are structured as hierarchical bindings. By doing so, the QoS mapping and monitoring processes become distributed from end-to-end, enhancing scalability and avoiding problems of centralised control. The hierarchical management approach is also suited to the engineering of adaptation mechanisms. At the upper levels of the structure, adaptation mechanisms are responsible for coarse-grained actions, including reconfiguration and change of service. Descending towards the leaves of the hierarchy, adaptation mechanisms become finer-grained, usually in the form of atomic resource control. The processes responsible for the adaptation actions are maintained within binding components (bindings are simply communication abstractions between one or more potentially distributed object interfaces). Furthermore, each individual binding is responsible for maintaining, through monitoring and adaptation, the point-to-point QoS characteristics which are defined by its interfaces.

3. Core Distributed Resources

The term 'resource' is inherently vague. Resources can exist at varying levels of abstraction. For instance at the highest level the term could include an end-system or a network node. At a lower level, a resource could represent a physical device or some system wide shared resource such as a network connection or access to a physical disk. However, we believe that there is a finite set of resources, in the end-system and the network, in terms of which all other resources can be described, and that at least to begin with, we should concentrate on monitoring this limited set of resources and make adaptations accordingly.

In order to successfully share resources across distributed applications and, furthermore, offer sufficient guarantees on their availability (an obvious requirement for time critical continuous media applications), resources need to be scheduled. Scheduling is the process of determining the availability of resources at a particular instant in time. Through resource reservation, an application can request resources and in return the system can determine whether sufficient resources are available to service the given request.

Scheduling algorithms can only be effective if they are used in advance of the admission of the resource. However, the time scale between resource admission testing and admission is dependent upon the scheduling algorithm. Many algorithms, of varying complexity and usually focused at either the network or the end-system, have been suggested [Hyman, 95]. However, in scheduling resources for multimedia applications the use of exhaustive or statistical techniques is often inappropriate (applications such as VoD can be statically analysed, but dynamically changing applications such as video conferencing cannot). It is suggested that simple heuristic scheduling techniques are preferable, offering a low processing overhead, and thus being more suited to resource requirements which vary in real-time. A basic model for resource reservation is offered by [Wolf, 95]. Reservation in advance does require that the duration of the reservation can be calculated *a priori* and that the resource usage is scheduled from the reservation request. There are techniques, such as partitioning, which can be used to couple with non-advance resource reservation systems; however we feel that the reservation in advance scheme is suitable for our purposes.

4. Adaptive Resource Management

Many of today's applications continuously adjust their resource requirements. This dynamic nature is particularly evident in distributed multimedia applications which demand strict levels of QoS. Furthermore, the problem is intensified in general purpose distributed environments because resources, in the network and the end-system, must be shared across multiple contending applications. Some operating systems, especially real-time systems, employ resource scheduling techniques (as previously discussed) in an attempt to alleviate the problem. Reservation and admission does allow a system to offer firm guarantees provided that the resource can be guaranteed, e.g. wireless communication bandwidth cannot always be guaranteed. However, many applications, such as distributed games, have varying resource requirements which cannot be predicted. One solution is to over-book resources and hope that the application does not demand resources beyond those made in the reservation. This is not an ideal solution as it is likely to result in the inefficient use of resources. So what is the rationale behind the use of resource reservation and admission techniques in a general purpose operating system at all? Why not simply rely directly upon monitoring and adaptation? In response, we suggest that compared with adaptation processes, resource reservation and admission processes are relatively lightweight and their processes demand little of the system. We propose the use of a two-tiered system, using a form of provisional reservation and admission to support a best-effort management of resources. In the event that an application's resource requirements do change, then adaptation techniques are employed. Thus, the resulting system combines the benefits of resource reservation and admission with the flexibility of monitoring and adaptation.

4.1 Monitoring and Adaptation

Monitoring is the process of observing the utilisation of resources and/or QoS characteristics in the system. It is the responsibility of the monitoring process to observe events and provide messages indicating the occurrence of QoS contract violations. There are two approaches to monitoring, *intrusive* and *non-intrusive*. Intrusive monitoring means that the monitoring process takes periodic samples of resource availability and utilisation; this in turn means that resources are consumed by the monitoring process itself, an overhead which is sometimes unacceptable. Alternatively, monitoring processes can rely upon indications of events which are not within the bounds of an agreed QoS contract.

To structure our model, we now define the mechanisms required to support QoS adaptation in a distributed environment. Many resource fluctuations in a distributed system can be handled implicitly by the processing entities themselves. For example, a video decoder which receives a burst of frames and cannot process these before the next frames are expected may decide to drop a portion of the frames from the burst. In such a case, because the adaptation is very fine grained, the likelihood of the adaptation process being noticeable to the end user is small. However, more sizeable fluctuations may become apparent through resource monitoring and prediction as previously discussed, and given that a system monitoring process has indicated that QoS degradation is occurring, or QoS failure is imminent, ideally the system should adapt its resource utilisation in an attempt to maintain the end-user's level of service.

We identify three classes of adaptation mechanisms: *resource control*, making fine grained adjustments to individual resources in the distributed system; *reconfiguration*, altering the topology of the end-to-end processing; and *change of service*, allowing the user to prioritise services and adjust as necessary. The majority of adaptation mechanisms fit into one of these classes. Each mechanism has a certain granularity, and each may affect the resource distribution in a slightly different manner. The choice of adaptation mechanism in response to received monitoring indications, is dictated by a set of system-wide *adaptation policies*. An adaptation policy describes, in conjunction with the applications resource capacity regions, what adaptation processes (control, reconfiguration or change of service) should be executed in response to various QoS scenarios. Furthermore, policies can be used in conjunction with the previously discussed benefit functions, allowing the prioritisation of adaptation mechanisms. Policies are applicable from end-to-end and are associated with any of the core distributed resources and their processing configuration. We suggest that the specification of policies should employ open interfacing techniques, aiding extensibility and readily understood programming level abstractions.

5. Conclusion

We have proposed a general model for the support of QoS-based adaptation and resource management in distributed multimedia systems. Our perspective is across the complete end-to-end co-ordination and control of network and end-system resources, supporting end-to-end QoS constrained processing and communications.

We have also made progress towards the implementation of our proposed QoS adaptation mechanisms, using the general model of QoS adaptation in the development of the prototype DRMA which provides a distributed platform for the development and deployment of QoS-constrained continuous media applications. Our current work is directed towards a complete implementation of the DRMA together with further support for QoS management and QoS adaptation, in both the network and end-system bindings.

6. Acknowledgements

We would like to acknowledge the kind support of BT Labs in funding this research under their Management of Multi-service Networks University Research Initiative (BT-URI).

7. References

[Chatterjee, 97] S.Chatterjee, J.Sydir and B.Sabata, "Modeling Applications for Adaptive QoS-based Resource Management", Proceedings of the 2nd IEEE High Assurance Systems Engineering Workshop, Bethesda, Maryland, August 1997.

[Dang, 95] F. Dang Tran and V. Perebaskine, "TORBoyau: Architecture et Implementation", General Leclerc, Issy-les-Moulineaux, France, December 1995.

[Hyman, 95] J.M.Hyman, A.A.Lazar and G.Pacifici, "Real-time Scheduling with QoS Constraints", IEEE Journal on Selected Areas in Communications, Vol. 9, p1052-1063, September 1991.

[Nahrstedt, 95] K.Nahrstedt, A.Hossain and S. Kang, "Probe-based Algorithm for QoS Specification and Adaptation", University of Illinois, QoS Workshop, Paris, 1995.

[ISO, 97] ISO/IEC JTC 1/SC21, "Quality of Service in ODP – Attachment 1", January 1997.

[Waddington, 97] D.G.Waddington, C.Edwards and D.Hutchison, "Resource Management for Distributed Multimedia Applications", Second European Conference on Multimedia Applications, Services and Techniques (ECMAST '97), Milan, Italy, May 1997.

[Wolf, 95] L.Wolf, W.L.Delgrossi, R.Steinmetz, S.Schaller and H.Wittig, "Issues of Reserving Resources in Advance", Proc. 5th Intl. Workshop on Network and Operating System Support for Digital Audio and Video, Durham, New Hampshire, April 1995