A Normative Approach to Capture and Analyze Quality of Service Requirements of Distributed Multimedia Systems

Mangtang Chan[1], Kecheng Liu[2]

[1] Department of Computer Science, City University of Hong Kong,
83 Tat Chee Avenue, Kowloon, Hong Kong SAR, PRC

[2] Informatics Research Centre, University of Reading,
Whiteknights, Reading, Berkshire, RG6 6AY, United Kingdom

Abstract. With the availability of powerful hardware, it is now possible to manipulate multimedia data with normal desk top computers. Further enhanced by the World Wide Web, the Internet serves as a platform for truly distributed multimedia systems (DMS). Non-functional requirements play an important role in the analysis and design of DMS and one of them is quality of service (QoS). This paper proposed a methodology to integrate QoS modeling under a semiotic framework which can then be used to analysis both the functional and non-functional requirements of DMS. The semiotic framework is agent-oriented. QoS characteristics and requirement for agents would first be defined, a normative approach would then be used for static model checking, admission test as well as run-time QoS monitoring and policing. The methodology is demonstrated by examples of some common DMS and related work in QoS modeling and specification will also be reviewed.

1 Introduction

Until recently, multimedia systems have been specialized applications operated with a substantial budget and usually used by special users in a special application domain. This is because of their characteristics of high demand in hardware capabilities and large network bandwidth. Together with powerful desk top computers and the World Wide Web, the Internet puts truly distributed multimedia systems (DMS) in the hands of nearly all types of users as long as they can have access to a browser. In designing systems, there are broadly two types of requirements, the functional and non-functional requirements. The functional ones represent the application features with which some useful work to be done and the non-functional ones determine how well the functional ones are delivered in terms of user experience. A major portion of these non-functional requirements can be grouped under the notion of quality of service (QoS). Analysis and design methodologies in the past did not put emphasis on these issues. These requirements would usually be handled after their functional counterparts. Treating QoS as an afterthought would not work for distributed multimedia

Chan M. and Liu K. (2005).
A Normative Approach to Capture and Analyze Quality of Service Requirements of Distributed Multimedia Systems.
In *Proceedings of the 1st International Workshop on Requirements Engineering for Information Systems in Digital Economy*, pages 4-13
DOI: 10.5220/0001423200040013

systems. As in other emerging fields of studies, there have been different works done for QoS employing different methodologies. These works could be generally categorized as understanding and defining QoS through surveys [16], [13] and case studies [2], [17]; applying or extending existing methodologies to cater for the need of QoS [5], [15], [1] and designing new specification languages or models [4], [6]. This paper proposes a normative approach under the semiotic framework [9] to capture and analyze QoS requirements. The main contribution is the integration of both functional and QoS requirements in the same model for analysis and design. Next sections are organized as follow. Related work in QoS will be briefly reviewed in section 2. In section 3, the modeling of DMS using the semiotic framework will be introduced. The normative approach of QoS requirement capture and analysis will be described in section 4 which will then be followed by the conclusion.

2 Related Work

Various terms and concepts have been clarified in a survey by [16] in which QoS was defined as "quality of service represents the set of those quantitative and qualitative characteristics of a distributed multimedia system necessary to achieve the required functionality of an application". Different QoS parameters of all components forming a distributed multimedia system, the so-called end-to-end QoS [14], have been discussed. The parameters would be derived for components from network, communication protocols, operating systems, databases and file servers, up to user interfaces and end systems. Related to QoS parameters, the ETNA project [12] has provided a comprehensive taxonomy of QoS requirements for different media as well as situations in which the media are used. In the project's terminology, QoS requirement would be different for video being used in a foreground or background mode for the purpose of telepresence or teledata. The taxonomy could serve as a check list for designers not to miss any relevant parameters during analysis. ITU-T recommendation X.641 [8] has proposed a QoS framework to describe how QoS can be characterized and how QoS requirements can be specified and supported by QoS mechanisms. QoS mechanisms are selected and configured according to QoS specification, resource availability and management policy.

Lots of research work in QoS using different approaches have been done in recent years, some examples are: [13] modeled QoS characteristics using the Djinn framework; [10] used meta-data stored in database or as extension of HTML to perform QoS management in the World Wide Web; [17] used an Application Programming Interface (API) approach to extend the Java architecture for design and implementation of QoS-aware applications; and [5] developed a QoS oriented transport protocol with UML-SDL (Unified Modeling Language – Specification and Description Language) modeling. UML has gained a lot of attention as a de facto standard for object oriented design. It has been extended through the use of profiles for more specialized application domains such as real time and fault tolerant areas. [4] also introduced XQoS, a XML-based mark up language for QoS specification. Although these different approaches have not converged into a mainstream or standardized approach,

two general trends could be observed, they are the design of QoS specification languages and the use of UML QoS Profile [15] to model DMS. Examples of specification languages are QML [6], CQML [1] and HQML [7]. Our approach differs from the specification language approach mainly in the integration aspect. Most specification languages address QoS in a specific and independent way and do not work under an integrated design and analysis methodology. UML QoS Profile has some degree of integration but mainly provides annotations and explicit run time QoS monitoring may not be specified clearly.

3 DMS Modeling Under a Semiotic Framework

The semiotic framework used to model DMS was described in detail in [3]. The framework is agent-oriented and takes a normative approach to define the actions of agents. Analysis is done in 3 main steps : identification of semantic units or vocabulary of the system by building an ontology; semantic analysis producing ontological charts which identify agents, their capabilities (affordance) and the relationships among agents and affordance (ontological dependencies); and finally, the definition of rules (norms) governing the behavior of agents. The ontology of semantic units in DMS also defines the generic-specialized and part-of relationships among semantic units. Figure 1 shows a simplified ontology for DMS. A system consists of one or more nodes, which are interpreted as computer systems at a certain location. If the system has one node, it is a multimedia system involving one computer e.g. a desk top computer or a DVD player. A number of nodes connected together in different locations would then become a DMMS. A node consists of one or more devices. Speaker and screen are device types. A node also consists of, or more precisely, supports the operations of some media types, which could be video, audio or text, etc.

Ontological dependence relationships of some common scenarios of DMS are illustrated in figure 2. Both the ontology and the ontological charts are visual tools to help in analyzing the system to be built and the results will be stored in a semantic database which can be used both in design time as well as run time. The conventions in drawing up ontological charts are circular shape represents an agent and rectangle represents an affordance. The affordance is always located at the right hand side of the agent to signify the relationship that the existence of an affordance depends on the agent.

It is possible to have more than one affordance depending on an agent or on a series of affordances. A textual format can also be used to represent the relationship such that

A(x) - A is the agent and x is the affordance that A is capable of

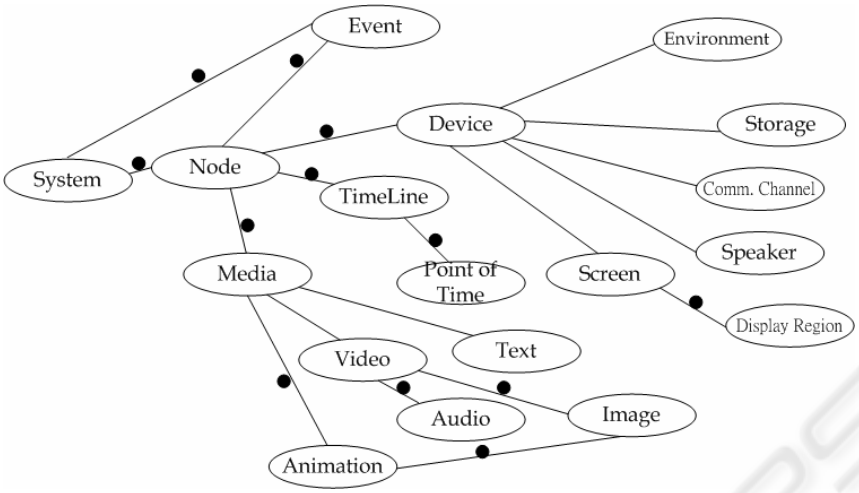

Fig. 1. Part of the ontology for DMS

It should be noted that A(x) itself is also an agent, a modified one, and A(x)(y) will become another agent who will be capable of doing y only when A possesses affordance x. Figure 3 shows this relationship diagrammatically for a video playing agent that is capable of playing MPEG format. The MPEG playing agent can be further modeled as three low-level I, B and P frame processing agents. Using the above textual format, the representation is then

video(play)(MPEG)
video(play)(MPEG)(I frame)
video(play)(MPEG)(B frame)
video(play)(MPEG)(P frame)

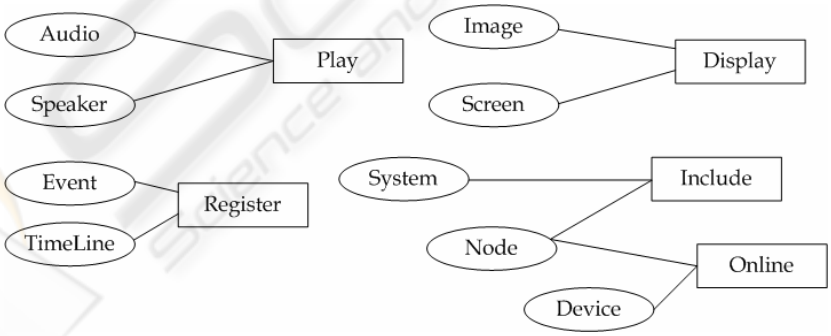

Fig. 2. Connection of agents through affordance

It is up to the analyst to determine the level of abstraction and specification of all required agents and their affordance would define structure of the application. The

behavior of the application will be determined by the collective behavior of the agents which are governed by rules known as norms. Norm has a general format of :

If <condition> then <D> <agent> <action>

D is an deontic operator [11], can be one of the following : obligatory, permitted and prohibited. A possible example is If <running interactively> then <permitted><video playing agent><ask for user preference of resolution>

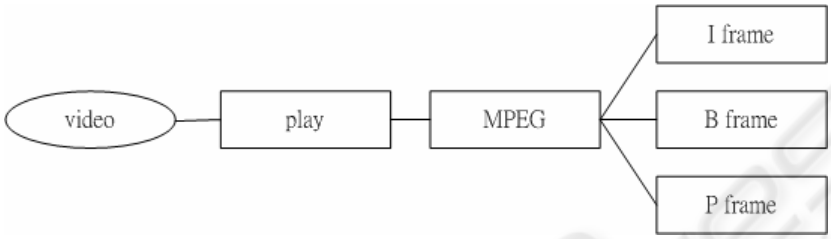

Fig. 3. Ontological dependence of a video playing agent

4 QoS Specification Under the Semiotic Framework

4.1 QoS Characteristic

According to the ITU-T recommendation X.641 [6], QoS modeling involves specification of QoS characteristics, QoS requirements and the mapping of requirement to QoS constraints of services. To integrate QoS modeling in the semiotic framework, QoS characteristics are specified as properties of agents while the QoS requirements and QoS constraints are specified as norms. Taking an example from [11], a presentation of a Web page consists of an audio stream, a video stream and an image to be processed, the application handling this presentation can be modeled as :

audio(play)
video(play)
image(play)

The QoS characteristics are specified as properties of each of the agent as follow:

audio(play)# QoS characteristic# bandwidth = 32 kbps
video(play)# QoS characteristic# bandwidth, GetBandwidth
image(play)# QoS characteristic# reliability = partial

The above syntax of the first property specification defines an attribute bandwidth of the type QoS characteristic for agent audio(play) and its value is now known at design time and is set to 32kbps. The attribute bandwidth for agent video(play) is defined but its value is not known at design, its value has to be defined during run time. There should be a corresponding affordance to obtain the valued at run time, i.e. video(play)(GetBandwidth). Another attribute reliability is also defined for agent image(play) and its value is known at design time to be partial.

4.2 QoS Requirement

QoS requirement can be regarded as the management of a QoS characteristic. The QoS requirement for the video stream of the presentation can be specified at design time or at run time as follow:

video(play)# QoS requirement# bandwidth, bandwidth >= 342 kbps
video(play)# QoS requirement# bandwidth, GetBandwidth(bandwidth)

It should be noted that the QoS requirement known as bandwidth is defined by a logical operation of the QoS characteristic bandwidth and a QoS requirement always has a corresponding QoS characteristic. Sometimes the QoS requirement could be too complicated to be represented by a simple predicate, it is therefore flexible to allow for a function of the corresponding QoS characteristic to be specified, e.g. 99% of the response time should be below 1 second. QoS requirement in the form of a statistical value such as a probability distribution can also be specified in this way. As illustrated in Figure 2, it is also possible to break down the QoS analysis for the video player into more detail. If the application is required to analyze individual video frame type processing, the QoS specification could then be defined as :

video(player)(MPEG)(I frame)# QoS characteristic# bandwidth = 75 kbps
video(player)(MPEG)(P frame)# QoS characteristic# bandwidth = 138 kbps
video(player)(MPEG)(B frame)# QoS characteristic# bandwidth = 129 kbps

Each of the frame processing may have different QoS requirement, e.g. :

video(play)(MPEG)(I frame)# QoS requirement# reliability = full
video(play)(MPEG)(P frame)# QoS requirement# reliability = partial

4.3 QoS Constraint

QoS constraint some times is used interchangeably with QoS requirement in some of the QoS research. It is used with a distinct semantic in our model. QoS constraints could be thought of being imposed on an agent by a service, e.g. if an agent sends data over a network, the service provided by the network would impose certain constraints on the data transmission because of the bandwidth or other factors. Depending on how the application is modeled, the service and in turn the QoS constraint can

be coming from the environment in which the agent resides or the service is provided by another agent. In the former case, the QoS constraint will be defined in the agent asking for the service, may be together with QoS requirement. This could be done if the designer chooses not to model the service explicitly and let the agent to interact with the environment to find its course. For the latter case, the QoS constraint will be specified in a different agent from the one asking for the service. Specification of the QoS constraint will be similar to the QoS requirement :

video(play)# QoS constraint# bandwidth = 342 kbps
video(play)# QoS constraint# bandwidth, GetBandwidth

The run time definition of QoS constraint is particularly useful as most of the time, the constraint would not be known at design time or it is dynamic in nature and will keep on changing during the life time of the application. If the QoS constraint is known at design time, the designer can determine whether the service can be used right away. This type of static checking would be useful for initial model checking even before prototyping is done. As a semantic database will be used under the semiotic framework, assistance is available to the designer by searching for appropriate agents from previous designs to provide the necessary QoS constraint.

Figure 4 depicts the case where two agents form a relationship of service requester and provider. The video playing agent has a role of source and transmits the video to a network agent having the role of destination. This relationship and definitions of QoS requirement and constraint can be represented as :

(video(play)#role source, network#role destination)(transmit)
video(play)(transmit)#QoS requirement bandwidth >= 342 kbps
network(transmit)#QoS constraint bandwidth, GetBandwidth

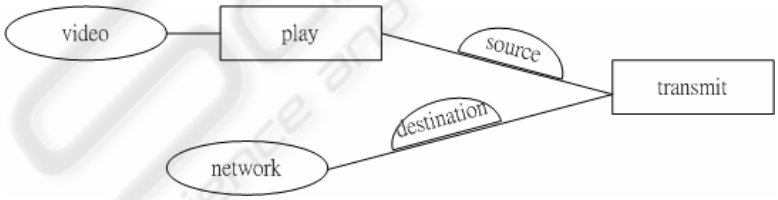

Fig. 4. Ontological dependence showing service requester and provider relationship

4.3 QoS Norms

After the specification of QoS requirement and constraint, the next of step of QoS analysis is to determine various actions according to the mapping of the requirement to the constraint. The specification of actions will be in the form of QoS norms, rules

that deal with QoS analysis specifically. Other norms governing non-QoS behavior of agents may, of course exist. A general format for QoS norm is :

if <QoS requirement Satisfied By QoS constraint>
then <permitted> <agent><action>

There are a number of situations in which evaluation of QoS constraint against QoS requirement is necessary. One situation has been mentioned is the static model checking, other cases are admission test and QoS monitoring or policing. Different actions would be initiated based on the result of QoS analysis and the agent state. Referring to the previous example of the Web page presentation, the following QoS norms can be set up :

if <bandwidth satisfied by bandwidth)>
then <permitted> <video(play)> <to instantiate and start running>

This is an admission test performed when the agent video(play) instantiates. The bandwidths represent QoS requirement and QoS constraint of the agent video(play) respectively. If the requirement is met, i.e. the requirement is satisfied by the constraint then video playing can start or else the playing has to wait. The requirement and constraint can be coming from the definitions of the same agent, or if the agent has a role defined, the constraint will be coming from the role partner and there will be no ambiguity. The QoS norm can further be generalized to specify actions for different situations, some examples are :

whenever <bandwidth not satisfied by bandwidth>
if <instantiation>
then <probibited> <video(play)> <to start running>

whenever<bandwidth not satisfied by bandwidth>
if <running>
then <prohibited> <video(play)> <to continue>

whenever<bandwidth not satisfied by bandwidth>
if <instantiation>
then <permitted> <video(play)> <ask user to accept lower frame rate>

whenever<bandwidth not satisfied by bandwidth>
if <running>
then <obliged> <video(play)> <to take corrective measures>

The first and second norms define a situation where QoS is guaranteed, if requirements are not met, the service is not admitted and if it is running, an abortion has to be done. The third and fourth norms indicate QoS is provided with the best effort, application is allowed to carry on but with less quality and without guarantee. User intervention can happen and the corrective measures could be frame dropping.

5 Conclusion

This paper proposes a methodology to integrate QoS specification and analysis into the semiotic framework for distributed multimedia systems modeling. This approach can put functional and non-functional requirements together for analysis and design. Both static model checking and dynamic QoS monitoring can be supported. In comparison, the approach of using a QoS specification language only focuses on the QoS but not the application requirements. Use of UML QoS profile, on the other hand, can add annotation to functional UML models but the QoS mapping and monitoring cannot be visualized easily. Specification of QoS characteristics, requirements and constraints could be done in flexible ways, including static values, range of values or run time dynamic evaluation. Specification of QoS norms provides a clear definition for actions to be taken during QoS admission test and policing. It also fits in well with the agent-oriented architecture. Although implementation consideration is outside the scope of this paper, building QoS aware applications using agent technology together with a rule engine would be feasible. Implementation investigation for models designed under the semiotic framework will be the next extension of this paper.

References

1. Aagedal Jan Oyvind, Ecklund Jr Earl F, 2002, Modelling QoS : Towards a UML Profile, Proceedings of UML 2002, Dresden, Germany, Springler Verlag LNCS 2460, pp.275-289.
2. Bernardi Simona, Petriu Dorina C, 2004, Comparing two UML Profiles for Non-functional Requirement Annotations : the SPT and QoS Profiles, Proceedings of the Workshop on Specification and Validation of Real-time and Embedded Systems, SVERTS 2004, Lisbon.
3. Chan Mangtang, Liu Kecheng, 2004, Semantic Analysis and Dynamic Representation for Collaborative Design of Distributed Multimedia Systems, Proceedings of the 8th International Conference on Computer Supported Cooperative Work in Design, Vol. 2, pp.197-202.
4. Exposito Ernesto, Gineste Mathieu, Peyrichou Romain, Senac Patrick, Diaz Michel, 2003, XQOS: XML-based QoS Specification Language, Proceedings of MMM'03, The 9th International Conference on Multi-Media Modeling January 7-10, 2003, Taiwan, pp.114-134.
5. Exposito Ernesto, Senac Patrick, Diaz Michel, 2004, UML-SDL modelling of the FPTP QoS Oriented Transport Protocol, Proceedings of the 10th International Multimedia Modeling Conference (MMM 2004), 5-7 January 2004, Brisbane, Australia, pp.153-160.
6. Frolund Svend, Koistinen Jari, 1998, QML: A Lanuage for Quality of Service Specification, HP Laboratories Technical Report, HPL-98-10.
7. Gu Xiaohui, Wichadakul Duangdao, Nahrstedt Klara, 2001, Visual QoS Programming Environment for Ubiquitous Multimedia Services, Proceedings of IEEE International Conference on Multimedia and Expo2001(ICME2001), Tokyo, Japan, Aug. 22 - Aug. 25, 2001.
8. ITU-T, 1997, Information Technology - Quality of Service Framework, ITU-T X.641 (ISO/IEC IS13236).
9. Liu Kecheng, 2000, Semiotics in information systems engineering, Cambridge, U.K., Cambridge University Press.
10. Madja E., Hafid A., Dssouli R., von Bochmann G., Gecsei J., 1998, Meta-data modelling for quality of service (QoS) management in the World Wide Web (WWW), Proceedings of

the MMM'98 Multimedia Modeling, 12-15 October, 1998, Lausanne, Switzerland, pp.223-230.
11. Meyer J-J. Ch., Wieringa R. J., 1993, Deontic Logic: A Concise Overview, In: Deontic logic in computer science : normative system specification, J. Wiley (ed.) Meyer John-Jules Ch., Wieringa Roel J, pp.3-16.
12. Mullin Jim, Jackson Matthew, Anderson Anne H, Smallwood Lucy, Sasse Angela M, 2002, The ETNA Taxonomy, Report of the Evaluation Taxonomy for Networked Multimedia Applications Project.
13. Naguib Hani, Kinberg Tim, Mitchell Scott, Coulouris George, 1998, Modelling QoS Characteristics of Multimedia Applications, Proc. 13 th IEEE Real-Time Systems Symposium (RTSS '98), Madrid, Spain, December,
14. Nahrstedt Klara, 1995, End-to-end QoS guarantees in networked multimedia systems, ACM Computing Survey, Vol. 27 No. 4.
15. Object Management Group, UML Profile for QoS Fault Tolerance, http://www.omg.org/cgi-bin/apps/doc?ptc/04-09-01.pdf
16. Vogel Andreas, Kerherve Brigitte, Bochmann Gregor von, Gecsei Jan, 1995, Distributed Multimedia and QoS: A Survey, IEEE Multimedia, Vol. 2 No. 2 pp.10-19.
17. Zeadally S, 2000, Implementation and Performance of QoS-aware Java Applications over ATM Networks, The Computer Journal, Vol. 43 No. 4, pp.266-273.