The Quality of Service (QoS) Binding Model

#*Jan de Meer*, &*Abdelhakim Hafid*, **Arno Puder*

#*GMD-FOKUS, System-Engineering Research,*
Kaiserin-Augusta-Allee 31, D-10589 Berlin, Germany, e-mail: jdm@fokus.gmd.de

&*Advanced Communication Engineering Centre, University of Western Ontario,*
London, Canada, N6A 5B9; e-mail: hakim@csd.uwo.ca

**International Computer Science Institute,*
1947 Center St., Berkeley, CA, 94704 USA; e-mail: puder@ICSI.Berkeley.EDU

1. Architectural Approaches and their Capabilities to QoS Binding

In this paper we first introduce the concepts and components of TINA and CORBA which are related to QoS and binding. Then, we introduce certain "points of QoS concerns". These points represent specialized flow behaviour to show the causality chain of QoS control operations to be instantiated during binding. Elements of the binding procedure and the points of QoS concerns are represented by a suggested modelling notation extending TINA's ODL.

The TINA consortium has adopted from the standardization organisation ITU-T the "Object Definition Language ODL" [TINA 96]. The purpose of ODL is described as follows: "ODL is used for the specification of systems from the perspective of the ODP computational viewpoint. Template specification of *operational interfaces, stream interfaces, objects* and *object groups* are provided with. ODL can be seen as an extension of OMG's *Interface Definition Language* (IDL) with additions to support the modelling of TINA's connection management architecture especially for the binding procedure.

The specification of a *stream interface signature* contains the aggregated flow specifications. Each of which consists of a flow name, the flow direction and the type of flow attributed with QoS parameters. For the specification of the flow direction only the specification identifiers *source* and *sink* are allowed. The flow type cannot be further constrained by e.g. QoS attributes because only a literal way of flow attributing exists.

A *QoS attribute* consists of a pair of a QoS type and of a QoS name. The QoS type for example identifies video or audio flow qualities and the QoS name represents literally the values of the QoS-type. For example, in order to attribute the audio and video flows of a stream, one could specify the audio flow QoS attribute as the pairs *audio-flow-type sound-quality* or *video-flow-type picture-quality* respectively. The first attribute can be read as "the *audio flow-type* must be of *sound quality* throughout the whole flow.

IDL is used to describe the interfaces to server objects and the signatures of operations provided by these interfaces, that client objects can call; interfaces are similar to classes in C++ and interfaces in Java. IDL is a declarative language which means that interfaces are defined separately from object implementations. Thus, objects might be implemented using different programming languages and still communicate with each other.

In order to deploy CORBA-based MM applications, CORBA has to provide means for QoS management. This is essential to satisfy the various QoS requirements for this type of applications. To make CORBA suitable for the development of MM applications, OMG defined a streaming mechanism (to handle audio and video) at the application level in the CORBA architecture [OMG 97]. The streaming mechanism defines a number of abstractions to deal with continuous media in multimedia applications. The abstractions are called *multimedia device, stream, stream endpoint, flow, flow endpoint* and *flow device*.

It is worth noting that several proposals have been made in the literature to extend CORBA with QoS capabilities to support MM applications [Laz 96, Sch 96, Cou95]. However, some of these proposals concentrate on the engineering level issues, while others do not support the "nice" abstraction proposed by OMG to support audio/video streams (e.g., stream and flow abstractions).

At the application level, we believe that more work/ extensions (to the proposed CORBA [OMG 97]) has to be done to support audio/video streams with regard to *QoS specification, QoS adaptation*, trade-offs between *QoS characteristics, QoS measurement*, and *cost*.

For example, the application programmer must be able to specify the application in terms of *probabilistic values of QoS parameters*; he/she should be able also to specify some policies, called *adaptation paths*, to be used when inevitable QoS violations occur.

More generally, it is up to the application programmer to decide about trade-offs between *QoS characteristics* for his/her application, e.g., the audio is more important than the video. These trade-offs will be used to define the adaptation paths when QoS violation occur. A scheme using the concept of importance has been used to compute *trade-offs* between QoS characteristics and cost in news-on-demand systems [Haf 97].

2. The Extended Binding Model for QoS

In TINA-ODL three types of interfaces are distinguished, i.e. the stream type, the operation type and the signal type. Whereas the stream type is of continuous nature, the two other ones are discrete. So far, a stream type interface can be described by the triple of *flow name*, *flow direction* and a *flow type*. All these descriptive notions are denoted by identifiers only. For the specification of the direction and of the type of a flow there exist reserved identifiers, e.g. *source, sink, audio, video* etc. This is insufficient for describing QoS-control functionality. Required notions are flow behaviour, flow observation, flow conformance statements and flow negotiation roles. Thus, the interface reference must be enriched with elements to define these QoS-types. Similar, the binding framework must be filled with the appropriate QoS type templates. Now, at binding these types will be instantiated by the actual behaviour and QoS characteristics. E.g. a bound observation point comprises the measurement operations for the QoS attributes. A bound conformance point comprises statements in order to compare observed QoS values with the contracted ones. A bound steering point comprises steering operations in order to influence the behaviour of the source of the flow.

The flow characteristics at each interface usually are not identical. Differences in the values of the characteristics exist, because the stream at each interface may flow with different speeds, and for the transmission of flows time will be consumed, thus the flow to be bound will be observable sequentially obeying some elapsed time in between.

By the point of observations *PoObs* locally observable events, e.g. sending or receiving, are e.g. measured according to their arrival times or their numbers of occurrences. At the point of conformance *PoCnf* at each interface the flow will continuously be compared to the contracted qualities which are specified by the QoS invariants *Qinvars*. For the measurements so-called continuous variables *Cvars* are defined, which keep the recently updated value of a QoS attribute; for example the arrival time of the last cell of a transmission, or the mean value of the throughput observed at the interface during a fixed period. The point of observation use timers and counters, i.e. measurement facilities captured by *Mfacs*. At the point of steering *PoSt* signals from the control path are received and inserted into the source flow process of the controlled interface. Notice, that the point of control is not captured by the interface specifications, because control is not a local function but an end-to-end one. Hence, the point of control is located at the binding object.

3. On the QoS Binding Notation

The purpose of a QoS binding notation, which we want to call Q_{uite} $ID_{ea}L$, is to provide means to model and to support binding of all the functionality of the introduced Points of QoS concerns. Hence, the binding notation is similar to an interface definition language but with improvements with respect to QoS. Q_{uite} $ID_{ea}L$ has been designed in the style of ODL [TINA 96]. One of the reasons is, TINA's ODL provides with the necessary structuring features for interfaces and provides already with some basic features to specify streams at interfaces.

The *flow* is a very central concept in the binding mode; its specification rules should be defined. A *flow-specification* is a pair of specifications *flow-spec QoS-types-spec* (I_1) of a flow together with types of QoS. At an interface, a flow denoted by the specification of the direction *flow-direction* (I_3) and of the specification of the type *flow-type* (I_4). The *flow-direction* is simply indicated by the keywords of *source* and *sink*. The specification of the *flow-type* is more complicated because it captures the behaviour of the flow at one side of the binding. A flow consists of two locations, one of which is located at the interface of the source of the flow and the other one at the interface of the sink of the flow. A continuous specification of a flow is represented by a recursive operational equation. The construction of a flow is achieved by a binary *flow-operator* by which single *flow-events* from the flow are operated. The flow is bound to the interface location *flow-location-name* (I_5). The *flow-event* is sub-structured into a *flow-element* and a *flow-measure*. The *flow-element* can be the specification of any unit of transportation, e.g. a frame on the application level, or a cell at the ATM-level or even a bit. The *flow-measure* supports the specification of *event-timer* and *event-counter.* The *flow-observation* realizes the "Point of Observation" of the binding model.

In the following we propose to employ a meta-notation for the binding notation Q_{uite} $ID_{ea}L$. A meta-notation is a "notation about notation". E.g., the Backus Naur Form (BNF) is an example of a meta-notation that can describe the grammar of context free languages. The purpose of using a meta-notation for the modelling language of QoS is

1. Notice in the current definition of TINA's ODL the *flow-type* is simply denoted by names. Multi-media streams can be distinguished by choosing names like *audio* or *video*. There are no means to express the continuous behaviour of streams.

to remain technology independent. Herewith, we follow the spirit of ODP which only seeks to provide a framework for open distributed systems, but not to prescribe any specific technology.

We will show how to use a *knowledge representation (KR)* technique as the aforementioned meta-notation. The *KR* technique we will use in the following is based on *Conceptual Graphs (CG)*. Conceptual graphs have been developed to model the semantics of natural language and to support the machine-based processing of natural language. Specifications based on conceptual graphs are therefore intuitive in the sense that there is a close relationship to the way human beings represent and organize their knowledge. From a mathematical point of view a conceptual graph is a finite, connected, directed, bipartite graph. The nodes of the graph are either concept or relation nodes. Due to the bipartite nature of the graphs, two concept nodes may only be connected via a relation node.

In doing conceptualizing one must define the domain specific "vocabulary" to be used for the concept and relation nodes to represent CG-based QoS specifications. As for the flow specification from rule I_5 above, this vocabulary might comprise concepts like *flow-event, flow-operator or flow-location-name*. Having done so, it is already possible to implement generic tools for the binding process as described in this paper. Once different notations like the CORBA-IDL or the TINA-ODL have reached the appropriate expressiveness to represent QoS specifications, one can define mappings to and from the CG notations. This has already been accomplished for the current version of CORBA-IDL and DCE-IDL [Pud 97].

The formal specification contains lambda expressions which provide further "explanations" for individual concepts. Part of the formal specification of the rules for flow specification is the lambda expression for the concept [operational-flow-type]. With the help of CG-based tools, the visualization of this looks as shown in the figure 1.

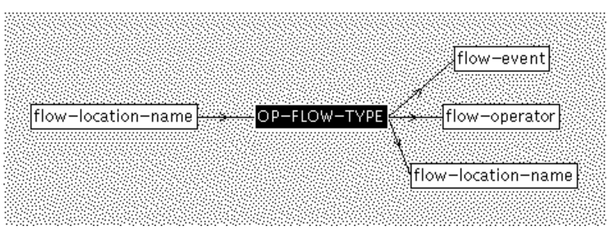

Fig. 1. An Example of the Utilization of CG-Based Tools

The white nodes denote concepts and the black node denotes a relation. The screenshot was taken with one of our CG-tools which already gives an idea of the kind of tools available for users in the handling of QoS constraints.

4. *Conclusions*

A QoS-extended binding model has been elaborated by augmenting the binding concepts of the ODP framework with the QoS points of concern, which comprise the declaration of continuous stream behaviour at end-points, observations, conformance, control, steering and negotiation. The suitability of the extended binding model has been validated by several examples, e.g. the specification of a video system which is bound by an ATM network. Unfortunately, the presentation of examples like these go beyond the scope of this paper, but can be read in other publications, i.e. [Jdm 98].

The QoS Binding Model provides a way of mapping QoS from an application level to the component and networking levels. The application level is where the objects of the application community are grouped and their interfaces are contracted. The negotiated QoS characteristics are mapped to the component level by the binding process which instantiates components accordingly. Hence, QoS mapping is part of the extended binding model. For example, the potential QoS of an ATM network, as it might be required by a video user, must be mapped to the observation and conformance points at the interface between the end user and the network access point.

Acknowledgement

We would like to thank Andreas Hoffmann from GMD-FOKUS who checked the QoS notational extensions against the actual ODL language definition of the ITU-T about consistency.

References

[Cou 95] G. Coulson et al. "Supporting the Real-time Requirements of Continuous Media in Open Distributed Processing", Computer Networks and ISDN Systems, Vol. 27, No. 8, 1995.

[Haf 97] A. Hafid and G.v. Bochmann, "An Approach to QoS Management in Distributed MM Applications: Design and an Implementation", Multimedia Tools and Applications Journal, 1997 (to appear)

[Jdm 98] Jan de Meer "Towards a QoS Binding Notation" to appear in proceedings of FDT'98, Cottbus, Germany, June 1998

[Laz 96] A.A. Lazar, K.S. Lim, and F. Marconcini "Realizing a Foundation for Programmability of ATM Networks with the Binding Architecture", IEEE JSAC, No. 7, pp.1214-1227, September 1996.

[OMG 97] Object Management Group RFP Submission "Control and Management of Audio/Video Streams", May 1997

[Pud 97] A.Puder, K.Geihs "Meta-Level Service Type Specification", Proceedings FIP/IEEE ICODP'97, Toronto, Canada, 1997

[Sch 96] D. C. Schmidt and A. Gokhale "The Performance of the CORBA Dynamic Invocation Interface and Dynamic Skeleton Interface over High-Speed ATM Networks", IEEE Globecom'96, London, UK. November 1996.

[TINA 96] A.Parhar (Editor) "TINA Object Definition Language Manual", Version 2.3, July 1996