Technical Report 356/07

Formalization of Network Quality-of-Service Requirements

Christian Webel[1] Reinhard Gotzhein[1] Daniel Schneider[2]

[1] Computer Science Department, University of Kaiserslautern
Erwin-Schrödinger-Str., D-67663, Kaiserslautern, Germany
{webel, gotzhein}@informatik.uni-kl.de

[2] Fraunhofer Institute for Experimental Software Engineering
Fraunhofer-Platz 1, D-67663 Kaiserslautern, Germany
Daniel.Schneider@iese.fraunhofer.de

Contents

Abstract

The provision of *network Quality-of-Service* (network QoS) in wireless (ad-hoc) networks is a major challenge in the development of future communication systems. Before designing and implementing these systems, the network QoS requirements are to be specified. Existing approaches to the specification of network QoS requirements are mainly focused on specific domains or individual system layers. In this paper, we present a holistic, comprehensive formalization of network QoS requirements, across layers. QoS requirements are specified on each layer by defining QoS domain, consisting of QoS performance, reliability, and guarantee, and QoS scalability, with utility and cost functions. Furthermore, we derive preorders on multi-dimensional QoS domains, and present criteria to reduce these domains, leading to a manageable subset of QoS values that is sufficient for system design and implementation. We illustrate our approach by examples from the case study *Wireless Video Transmission*.

1 Introduction

One of the major challenges in wireless (ad-hoc) networks is the provision of *network quality of service* (network QoS), *i.e.* the quality of service provided by the underlying communication system. The need for network QoS arises from the fact that, for state-of-the-art distributed user applications, it is essential, to offer their functionality with a certain degree of quality, which requires suitable communication mechanisms.

State-of-the-art wireless distributed communication systems must offer proactive and intelligent behaviour in order to cope with varying channel quality and connectivity. In other words, wireless communication systems must facilitate changes at runtime, and these changes are to be performed according to an effective reasoning about user, environment, and system context. The realization of such adaptive behaviour can in fact be seen as one of the technological key challenges in the development of wireless communication systems supporting *network quality of service*.

One of the main drivers of adaptive behaviour is the need to maintain specific non-functional properties, *i.e.* a specific level of QoS (Quality of Service) for the provided services. Especially in the wireless domain, where resources (like bandwidth, energy, processing power, and memory) are inherently scarce and subject to frequent change, the systems need to manage their resources in a QoS-aware way. To this end, to form a basis for corresponding adaptation mechanisms to work on, a major prerequisite is an explicit specification of QoS offers and correlated needs. Moreover, QoS as an inherently cross-cutting concern has to be considered from end-to-end and from user layer down to the hardware layer. The QoS specifications hence reside on different layers of abstraction and need to be *mapped* on each other.

Our current work aims at establishing a holistic engineering approach for wireless systems. As a central result, we envision a development framework for wireless systems that comprises among other things a *QoS requirement specification*. Because current QoS specification techniques are mainly focused on specific domains or layers of abstraction and not addressing the special characteristics of the wireless field like high dynamics or scarce resources, we have investigated the specific requirements for network QoS for a holistic QoS specification approach in [7]. Moreover, we correspondingly proposed a first general metamodel for QoS specifications. In this paper, we present a formalization of network QoS requirements incorporating adaptation policies and supporting the concept of QoS mapping. The core ideas of our metamodel are formalized in order to provide a foundation for the sound integration of these concepts into our holistic engineering approach.

The remaining part of the paper is organized as follows: In Section 2, we survey related work. Section 3 introduces network quality of service in the context of our approach. Sections 4, 5 and 6 describe the requirements on and the formalization and specification of network quality of services. In Section 7, we evaluate our approach. Last, we conclude with conclusions and future work.

2 Related Work

According to various requirements posed by different system designs, user preferences, middleware, hardware, networks, operating systems, and applications, numerous QoS specification techniques emerged over the last few years. A good overview on QoS specification techniques is given in [3]. Recognizing the importance of a taxonomy for QoS specification techniques, a broad range of existing QoS specification techniques according to their layer of abstraction and their characteristics are classified.

One of the best known QoS specification techniques is QML (Quality Modelling Language), which has been proposed by Frølund and Koistinen [1]. However, QML is focused on the specification of application layer QoS properties whereas in wireless systems it is also important to explicitly deal with hardware layer properties. A further widely recognized approach which considers both, application layer and resource layer properties, is QDL (Quality Description Language). QDL has been proposed as a part of the QuO (Quality Objects) framework [8] that supports QoS at the CORBA object layer. CQML [2] adopts some of the fundamental concepts of QML but overcomes its major shortcomings, as, for instance, lacking support with regard to dynamic adaptation. On the other hand, CQML is still too much focused on application layer QoS.

In [4], the authors describe an approach for specifying and mapping QoS in distributed multimedia systems. Based on the specification, fuzzy-control is used for QoS adaptation. We extend and formalize some basic ideas presented in [4] to fit to wireless networks.

3 Network Quality of Service

Network Quality of Service can be defined analogously to Schmitt [6] as

> *the degree of well-definedness and controllability of the behaviour of a communication system with respect to quantitative parameters.*

On hardware layer, typical parameters are, *e.g.*, throughput, delay, delay jitter, and loss. Depending on the application, the requirements on network QoS may vary. Bulk transfer applications like ftp transactions are more interested in the overall throughput, whereas real time applications rely on bounded delay and delay jitter.

The specification of network QoS requirements is a crucial part in the tailored development of communication systems. It precisely describes the needs of one service user on a communication system and therefore forms part of a traffic contract between service user and service provider, *i.e.* the communication system.

In the following sections, we will use our case study *Wireless Video Transmission* [9] for illustration purposes. It consists of one video data source and one video projection facility, interconnected via a wireless medium. There are three parameters to adjust the video source: video resolution, JPEG compression in percent as a quality factor, and frame rate. The need for adaptive QoS in a wireless environment arises from the heavy network load caused by video transmission. While in wired networks, the available bandwidth is usually sufficient, video transmission can absorb almost all available communication resources in unstable wireless ad-hoc networks, depending on the chosen video frame rate, image quality and resolution. Without a specification of the needed QoS and further mechanisms operating on the specification, this resource consumption may lead to a situation where the video transmission tends to congest the medium and thus prevents all wireless communication.

4 Types of QoS Requirements

In this section, we elaborate on the types of requirements a QoS specification has to address. These requirements can be directly derived in accordance with the definition of network QoS.

4.1 Performance Requirements

Performance describes the efficiency aspects of the QoS requirements characterizing the required amount of resources and the timeliness of the service (*e.g.,* peak and average throughput, delay, jitter, burst characteristics).

4.2 Reliability Requirements

Reliability describes the *safety-of-operation* aspects of the QoS requirements characterizing the fault behaviour of the service (*e.g.,* loss rate and distribution, corruption rate and distribution, error burstiness). It can significantly impact the overall throughput and functionality on lower system layers, since it introduces redundancy *e.g.* retransmission or error correction.

4.3 Guarantee Requirements

Guarantee describes the level of commitment characterizing the binding character of the service. It is described by one of four predefined categories:

- *best-effort*: The minimal type of commitment is called *best-effort.* The (remaining) resources are used to handle the user requirements in an unassured manner.
- *deterministic*: The maximal type of commitment is called *deterministic.* The required and assured resources are always provided, *i.e.* the needed quality of service is always fulfilled. It is not possible to permanently achieve this kind of QoS guarantee in wireless networks due to varying channel conditions and node mobility.
- *statistical*: A soft form of deterministic guarantees. The required quality of service is guaranteed with a certain probability (deterministic is equivalent to a probability of 1). This enables overallocation of resources and hence more simultaneous QoS data flows.

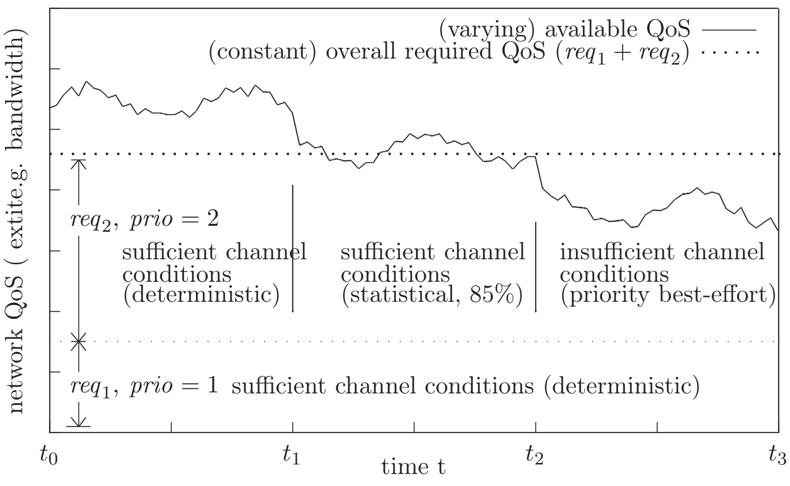

Figure 4.1: Enhanced Best-Effort

- *enhanced best-effort* [9]: *Enhanced best-effort* combines the three common commitments best-effort, statistical and deterministic in the following way: In periods of sufficient channel conditions (and stationary topology), *statistical* or even *deterministic* guarantees can be given. Otherwise, a *priority based best-effort* scheme is used. In summary, *better-than-best-effort guarantees* are given all times.

Figure 4.1 illustrates *enhanced best-effort* for two QoS requirements req_1 and req_2. In the interval $[t_0, t_1]$, the available network QoS exceeds the sum of both requirements, therefore, deterministic guarantees can be provided. In the period $[t_1, t_2]$, the available QoS is sometimes below the required QoS. In this case, deterministic guarantees for the requirement with the higher priority (req_1) can be given. The other requirement is served with statistical guarantees (85%). Finally, in the interval $[t_2, t_3]$, deterministic guarantees for req_1 are still possible, however, req_2 can no longer be supported with an adequate statistical guarantee, and therefore, priorities are applied to give preference to high-priority requirements (req_1) by reducing less preferential requirements (req_2). In summary, deterministic guarantees are given for req_1 all the time, whereas req_2 gracefully degrades.

4.4 Scalability Requirements

Varying communication resources often lead to an overload of the medium. To avoid this, the user data flow has to be adapted to the new resource situation. These adaption considerations, the control aspects of the QoS requirements characterizing the scope for dynamic adaptations to varying resource situations, are summed up in the description of *scalability*.

5 Formalization of Network Quality of Service

The need for *formalization* of network quality of service arises from the fact that a *precise* description of network QoS between service user and service provider is needed to police, control, and maintain the data flow a user emits to the communication system. Further on, the mechanisms realizing these functionalities need a *precise* and *well-defined* description of QoS. These mechanisms are typically integrated across layers, and therefore, more than one viewpoint on the required network QoS is needed. So another reason for formalization is the support for a well-defined translation of the specification between the different viewpoints on QoS, called *QoS mapping*.

In this section, we start to formalize network QoS by defining *QoS domain* and *QoS scalability*.

5.1 QoS Domain

A *QoS domain* captures all QoS characteristics of a class of data flows and incorporates the first three aspects described in the previous Section – performance, reliability, and guarantee. Hence, a *QoS domain* is the cartesian product of domains of *QoS performance* P, *QoS reliability* R, and *QoS guarantee* G,

Definition 1 (QoS Domain) *The QoS domain Q is defined as $Q = P \times R \times G$, where P is the performance domain, R is the reliability domain, and G is the guarantee domain. $q = (p, r, g)$ denotes an element of Q, called QoS value.*

5.1.1 QoS Performance

QoS performance describes efficiency aspects characterizing the required amount of resources and the timeliness of the service. The relevant aspects are included in the *QoS performance domain* P, which we formalize as follows:

Definition 2 (QoS Performance) *A QoS performance domain P is defined as $P = P_1 \times \ldots \times P_n = \prod_{i=1}^{n} P_i$, where $P_1, \ldots, P_n$ are performance subdomains.*

The performance domain P provides a traffic description for a given (user) data flow on a certain system layer. Therefore, performance domains for the requested and provided QoS have to be identified. On user layer, this is often done in a very abstract way

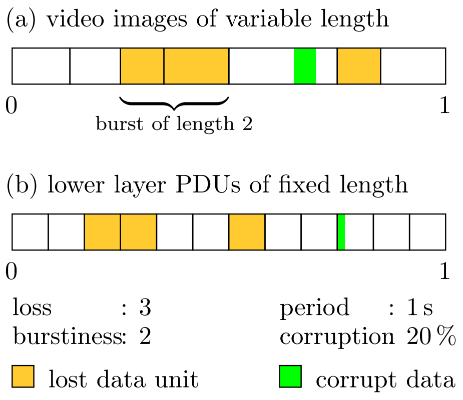

Figure 5.1: Impact of reliability parameters on different system layers

by distinguishing between different application scenarios, *e.g. surveillance* in case of a video transmission. On lower system layers, the level of abstraction decreases. *E.g.* on hardware layer, the performance could be described in terms of communication *bandwidth* and *delay*.

5.1.2 QoS Reliability

The relevant aspects describing the *QoSreliability* are included in the *QoS reliability domain R*:

Definition 3 (QoS Reliability) *The QoS reliability domain R is defined as $R = Loss \times Period \times Burstiness \times Corruption$, with $Loss = \mathbb{N}_0$, $Period = \mathbb{R}_+$, $Burstiness = \mathbb{R}_+$, and $Corruption = \{r \in \mathbb{R} \mid 0 \le r < 100\}$.*

Reliability addresses *data loss* corresponding to a layer-specific data unit (*e.g.* video images or lower system layer PDUs), the *period* in which data loss occurs (*e.g.* in seconds) and the *burstiness*, the duration of a successional appereance of data loss. Data loss in communication systems is typically caused by buffer overflows, delivering packets (data units) too late or, especially in wireless ad-hoc networks, changing topologies, link breaks, and interfering nodes. As a fourth parameter, the *corruption rate* for a layer specific data unit in percent is given. Even if *corruption* percent of the data unit is corrupt, the data unit may still be useful (*e.g.* if erroneous parts can be corrected).

Figure 5.1 illustrates the four parameters and their impact on different system layers. On a higher system layer, data loss is often specified in a more abstract way than on lower system layers, *e.g.* in video frames per second. Hence, a data loss of 3 frames per second may result in a larger percentage of loss than on lower system layers, where the data unit is a PDU of fixed size. The same holds for the corruption rate.

5.1.3 QoS Guarantees

The commitment aspect of the QoS requirements characterizing the binding character of the network QoS, the *QoS guarantee*, is formalized by the *QoS guarantee domain.*

Definition 4 (QoS Guarantee) *The domain of QoS guarantee G is defined as $G = DoC \times Stat \times Prio$, where $Stat = \{p \in \mathbb{R} \,|\, 0 < p \leq 1\}$, $Prio = \mathbb{N}$, and $DoC = \{bestEffort, enhancedBestEffort, statistical, deterministic\}$.*

The guarantee consists of a degree of commitment *LoS*, a corresponding statistical description *Stat* of the level of service and a *priority.* The priority determines the relative importance between two or more QoS requirements (traffic contracts). The level of service is selected from the four categories described in Section 4.3.

5.2 QoS Scalability

The *QoS scalability S* describes the control aspects characterizing the scope for a dynamic adaptation of the QoS aspects of a data flow (described by a QoS domain) to a certain granted network quality of service.

Definition 5 (QoS Scalability) *Let Q be a QoS domain. The domain of QoS scalability S is defined as $S = Util \times Cost \times Up \times Down$, where $Util = \{u \mid u : Q \rightarrow [0,1]\}$, $Cost = \{c \mid c : Q \rightarrow \mathbb{R}_+\}$, and $Up, Down \in \{x \in \mathbb{R}_+ \mid 0 \leq x \leq 1\}$.*

The elements of *Util* and *Cost* are called *utility functions* and *cost functions*, respectively. A utility function determines the usefulness of QoS values $q \in Q$ and therefore provides means to offer the service user a precise feedback of the currently granted QoS (see also [4]). This information is crucial for upscaling and downscaling, and has to be provided on all system layers. The utility of QoS values depends on the application scenario, but not necessarily on the amount of needed resources. The latter is expressed by the cost function, which can be tailored to the actual resource situation, associating higher costs with scarcer resources. In other words, given two QoS values q and q' with $u(q) > u(q')$, it is possible that $c(q) < c(q')$, i.e., q consumes less resources than q'. Related to the utility function, two values $up \in Up$ and $down \in Down$ are used to define thresholds for up- and downscaling, i.e. a scaling is only performed, if the benefit for the user increases/decreases more than $up/down$ percent.

According to [4], the utility function u can be built using three subfunction on P, R and G. The following equations define the subfunctions on the QoS domain $Q = P \times R \times G$

$$u_P : P \rightarrow [0,1], \quad u_R : R \rightarrow [0,1], \quad u_G : G \rightarrow [0,1] \tag{5.1}$$

A possible definition u for a QoS value $q = (p, r, g)$ is:

$$u(q) = \min\{u_P(p) \cdot w_P, u_R(r) \cdot w_R, u_G(g) \cdot w_G\} \tag{5.2}$$

This definition emphasizes that usually, a minimum benefit of each of the QoS value constituents is required. Other definitions can be given by introducing weights w_P, w_R, and w_G, reflecting the relative importance of performance, reliability, and guarantee, respectively, in the current application scenario, with $u(q)$ being the sum of the weighted constituents of q. In both cases, the result of (5.2) has to be normalized into the interval $[0, 1]$ (see Definition 5).

The utility function u (the cost function c) induces an *equivalence relation* $\sim_u$ ($\sim_c$) and a preorder $\lesssim_u$ ($\lesssim_c$) on the QoS domain Q:

$$\sim_u =_{\text{DF}} \{(q_1, q_2) \in Q \times Q \mid u(q_1) = u(q_2)\} \tag{5.3}$$

$$\lesssim_u =_{\text{DF}} \{(q_1, q_2) \in Q \times Q \mid u(q_1) \leq u(q_2)\} \tag{5.4}$$

In certain scenarios, several QoS values may have the same usefulness according to the utility function u. For instance, a user may not be able to distinguish between 25 and 26 picture frames per second, and therefore assigns the same utility value to both QoS values. For this reason, $\lesssim_u$ is a preorder on Q in general. Based on $\sim_u$ ($\sim_c$), we define u-equivalence (c-equivalence) classes of Q:

$$[x]_u = \{q \in Q \mid q \sim_u x\} \tag{5.5}$$

These definitions form the basis for consistency criteria of QoS mappings.

Apart from defining the utility of QoS values, the actual costs are required in order to provide the scope for dynamic adaptation. For instance, it is possible that for QoS values q, q', and q'', $u(q) > u(q') > u(q'')$, while the costs in terms of resources are $c(q') > c(q) > c(q'')$. Assume that q'' is currently provided, and the resource situation improves. In this case, it is certainly better to directly scale to q, omitting q'. This means that although q' has a utility in-between q and q'', it should not be used. This observation can be exploited such that for a given utility, the QoS value with minimum cost is selected. For each u-equivalence class, we keep one representative value with minimum cost (Step 1). Next, we observe that in general, while the utility increases, the cost may decrease. Therefore, some u-equivalence classes become obsolete, as it would be better to skip some QoS values to get even better utility for less cost (Step 2). These ideas are formalized in the following definitions.

To formalize Step 1 (keeping one representative per u-equivalence class with minimum cost), we define the *reduced* QoS domain Q^u by selecting the best element of each u-equivalence class of Q regarding c. Let m be the cardinality of $Q/\!\sim_u$, the quotient set of Q w.r.t. $\sim_u$, and let $[x]_u^i$ denote the ith element of $Q/\!\sim_u$ regarding $\lesssim_u$ (ith u-equivalence class). Then,

$$Q^u = \{q_1, \ldots, q_m\} \cap Q', \quad q_i = q \in [x]_u^i \mid \forall y \in [x]_u^i \cdot q \lesssim_c y, \quad 1 \leq i \leq m \tag{5.6}$$

Q^u contains elements in the specified subset Q' of a QoS domain Q (see Sect. 6, (6.1)) and is totally ordered by $\lesssim_u$.

To formalize Step 2 (discarding of QoS values with higher cost, but less utility), we define the derived QoS domain $Q^{u,c}$ as follows:

$$Q^{u,c} = \{q \in Q^u \mid \forall y \in Q^u \, . \, c(q) > c(y) \Rightarrow u(q) > u(y)\} \tag{5.7}$$

6 Specification of Network QoS Requirements

A *QoS requirements specification* is a concrete realization of the formalization of network quality of service. It is a set of valid QoS values and a description of the QoS scalability S.

Definition 6 (QoS Requirements Specification) *Let Q be a QoS domain and S be a QoS scalability domain. A QoS requirements specification qosReq is defined as a triple (q_{min}, q_{opt}, s), where $q_{min},\ q_{opt} \in Q$ and $s \in S$.*

The QoS values q_{min} and q_{opt} describe a subset $Q^{'} \subseteq Q$, providing information about valid elements of the QoS domain under the current QoS requirement specification. To obtain this $Q^{'}$, the preorder $\lesssim_u$ induced by the utility function (see (5.4)) is applied:

$$Q^{'} = \{q \in Q \mid q_{min} \lesssim_u q \lesssim_u q_{opt}\} \tag{6.1}$$

This is exemplified in Figure 6.1. In our case study *Wireless Video Transmission*, two QoS requirements specifications exist, *surveillance* and *panorama*, describing a video transmission serving *surveillance* purposes, and a landscape video recording (*panorama*). The QoS domain Q_{Video} describes the QoS characteristics of a video data flow on application layer. For both specifications, q_{min}, q_{opt}, and a corresponding scalability specification s are used to form the subsets $Q^{'}_{surveillance}$ and $Q^{'}_{panorama}$. On the domain Q_{Video}, two preorders $\lesssim_{u_s}$ and $\lesssim_{u_p}$ exist, defined by the corresponding utility functions of *surveillance* and *panorama*.

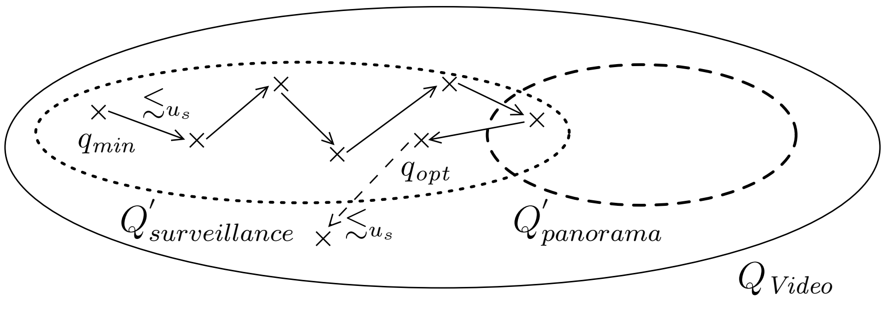

Figure 6.1: QoS Requirements Specification

6.1 QoS Domain

A specification of a QoS domain is done by specifying the three subdomains Performance P, Reliability R, and Guarantee G. After the specification, the domain has to be instantiated in order to form the QoS requirements specification subset Q'. In the following, this is done for the QoS domain Q_{Video} used for the *Wireless Video Transmission*.

6.1.1 QoS Performance

A typical element p of the performance domain P is called *performance requirements specification*, described by an *n-tuple* $p = (p_1, \ldots, p_n)$. So the first step when specifying the performance of a data flow is to identify and concretize the various performance domains describing the quantitative characteristics in a layer specific manner. The quality of video transmission depends on picture frame resolution, JPEG compression rate, and picture frame rate. Further QoS parameters are transmission delay and delay jitter, which we omit in the following. Hence, for our case study, the concrete domains on application layer are $P_1 = \mathit{Resolution}$, $P_2 = \mathit{Quality}$, $P_3 = \mathit{FrameRate}$ yielding P_{Video}.

$$\begin{aligned} P_{video} &= \mathit{Resolution} \times \mathit{Quality} \times \mathit{FrameRate} \\ \mathit{Resolution} &= \{(320, 240), (480, 360), (640, 480)\} \\ \mathit{Quality} &= \{25, 50, 75\} \\ \mathit{FrameRate} &= \{f \in \mathbb{N} \mid 1 \leq f \leq 25\} \end{aligned}$$

Typical element of P_{video} is $p = (\,(res_x, res_y), qual, fps\,)$. An appropriate specification of the required performance for surveillance purposes is given by

$$p_{minSur} = ((320, 240), 25, 10), \quad p_{optSur} = ((640, 480), 75, 20)\ .$$

6.1.2 QoS Reliability

The instantiation of the reliability domain R, the *reliability specification* r is a 4-tuple $r = (l, p, b, c)$. The reliability specification identifies concrete values for loss, period, burstiness, and corruption (see Definition 3). For the video transmission, we define

$$r = (3, 1, 2, 0)$$

specifying a permitted data loss of *three* picture frames per *one* second, loss bursts of up to *two* picture frames, and a corruption rate of *zero* percent.

6.1.3 QoS Guarantees

A *guarantee requirements specfcation* $g \in G$ is given by

$$g = (\mathit{enhancedBestEffort}, 0.8, 8).$$

If due to the current resource situation only priority best-effort guarantees are provided by the communication system, the priority of 8 enables the wireless video transmission to gain privilege over other applications with lower priorities (< 8). If the transmission system offers adequate statistical guarantees, a minimum of 80 percent of the required QoS is required.

6.2 QoS Scalability

To specify QoS scalability, concrete utility functions u_P, u_R, u_G, cost function c, and two thresholds *up* and *down* are to be defined. Thereto, we have to consider the application scenarios of both QoS requirement specifications. *I.e.* in case of $qosReq_{sur}$, the resolution and quality take priority over the video frame rate (surveilance), otherwise the video frame rate is the most important parameter (panorama). To specify the utility functions on P_{Video}, we define three auxiliary utility functions u_{res}, u_{qual} and u_{fps} operating on the three performance subdomains *Resolution*, *Quality* and *FrameRate* normalizing the utility of each parameter to a value in $[0, 1]$::

$$\begin{aligned} u_{res} &: Resolution \rightarrow [0,1], & u_{res}(res) &= \tfrac{res_x - 160}{480} \\ u_{qual} &: Quality \rightarrow [0,1], & u_{qual}(qual) &= \tfrac{qual}{75} \\ u_{fps} &: FrameRate \rightarrow [0,1], & u_{fps}(fps) &= \tfrac{fps}{25} \end{aligned} \tag{6.2}$$

Next, we define weights reflecting the relative importance of each subdomain corresponding to the current application scenario. For instance, the picture frame rate is the decisive video transmitter in case of surveillance, while resolution and quality are of particular importance in the panorama scenario. With the weights $\omega_{res} = 0.1$, $\omega_{qual} = 0.1$, $\omega_{fps} = 0.8$ for surveillance and $\upsilon_{res} = 0.1$, $\upsilon_{qual} = 0.1$, $\upsilon_{fps} = 0.8$ for panorama, we obtain the following performance utility functions:

$$\begin{aligned} u_{P_{sur}} &: P_{video} \rightarrow [0,1], & u_{P_{sur}} &= 0.1 \cdot u_{res} + 0.1 \cdot u_{qual} + 0.8 \cdot u_{fps} \\ u_{P_{pan}} &: P_{video} \rightarrow [0,1], & u_{P_{pan}} &= 0.4 \cdot u_{res} + 0.4 \cdot u_{qual} + 0.2 \cdot u_{fps} \end{aligned} \tag{6.3}$$

Since $r_{min} = r_{opt}$ and $g_{min} = g_{opt}$ within both specifications, the utility subfunctions operating on R and G can be defined as follows:

$$u_G(x) = \begin{cases} 0 & \text{iff } x < g_{min} \\ 1 & \text{otherwise} \end{cases}, \quad u_R(x) = \begin{cases} 0 & \text{iff } l/p > l_{min}/p_{min} \vee b \geq b_{min} \vee c > c_{min} \\ 1 & \text{otherwise} \end{cases} \tag{6.4}$$

u_G implies an order on the guarantee domain that can be intuitively given by arranging the values (1) according their *DoC* (*bestEffort* to *deterministic*), then (2) according the statistical component and last (3) according their priority. The parameters l, p, b, and c in the definition of u_R refer to loss, period, burstiness, and corruption, respectively.

With $w_p = w_r = w_g = \frac{1}{3}$ (see Section 5.2) we get the *normalized* utility functions u_{sur} and u_{pan},

$$u_{sur}(q) = \min\{u_{P_{sur}}(p), u_R(r), u_G(g)\}, \quad u_{pan}(q) = \min\{u_{P_{pan}}(p), u_R(r), u_G(g)\} \tag{6.5}$$

Table 6.1: $u_{P_{sur}}$-equivalence classes of P_{Video}

utility	$u_{P_{sur}}$**-equivalence class**
0.1	((320,240),25,1)
0.13	((480,360),25,1) ((320,240),25,2) ((320,240),50,1)
...	...
0.39	((320,240),25,10) ((640,480),75,6) ... ((480,360),25,9) ((480,360),50,8)
0.42	((640,480),25,9) ((480,360),25,10) ... ((320,240),25,11) ((480,360),75,8)
...	...
0.84	((640,480),25,22) ((640,480),75,20) ... ((480,360),25,23) ((320,240),75,22)
...	...
1.0	((640,480),75,25)

Apart from the description of the utility, we need another viewpoint on the QoS values, the *costs* in terms of *resource consumption*. W.l.o.g., we assume that the needed bandwidth of a video transmission would provide a good metric for the needed resources, leading to following cost function:

$$c(q) = \tfrac{(160 \cdot qual + 3000) \cdot (res_x - 160)}{160} \cdot fps \tag{6.6}$$

In both cases, downscaling should be performed if the benefit decreases by 5 percent and upscaling should only be done if the benefit increases by 10 resp. 5 percent, leading to the following complete specification of the scalability requirements

$$s_{sur} = (u_{sur}, c, 0.1, 0.05), \quad s_{pan} = (u_{pan}, c, 0.05, 0.05) \tag{6.7}$$

Based on the utility functions, the QoS values are divided into equivalence classes. Table 6.1 lists some $u_{P_{sur}}$-equivalence classes of P_{video}. In order to minimize the overall number of classes, the utility has been rounded to two decimal places, resulting in a reduction from 125 to 44 classes.

In Figure 6.2, the QoS domain is reduced, applying Steps 1 and 2 as defined in Sect. 5.2. The utility function u_{sur} partitions Q_{video} into 45 u_{sur}-equivalence classes (rounded to two decimal places). Since the result of u_R resp. u_G could be 0, the overall number of equivalence classes increases by one (cf. $u_{P_{sur}}$-equivalence classes). If all values of the QoS domain Q are arranged along the x-axis, respecting the preorder $\lesssim_u$, then the resulting graph is a monotonically increasing step function. In addition, the above cost function c describes the needed resources. Note that the costs basically increase with the utility, however, within a given u_{sur}-equivalence class, different costs may be associated with QoS values having the same utility. The reduced domain Q^u is formed by selecting the cost-optimal QoS values out of each u_{sur}-equivalence class (see Table 6.2) and intersecting this selection with $Q^{'}$, leading to $Q^u = \{q^u_{16}, \ldots, q^u_{39}\}$. Step 2 (cf. (5.7)) induces a further reduction of the overall number of QoS values, since for example q^u_{37} can be omitted due

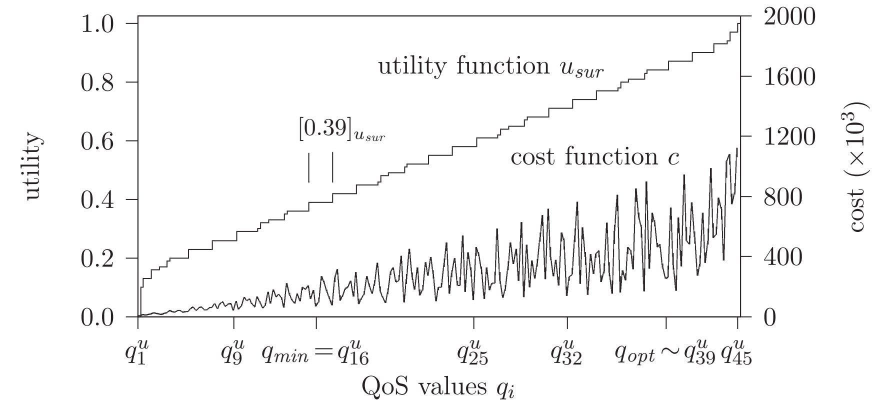

Figure 6.2: Reduction of Q

to the higher cost but less utility compared to q^u_{38}. This leads to $Q^{u,c}$ with a total number of 16 QoS values.

Table 6.2: Cost-optimal QoS values

utility	QoS value		cost
0.0	q^u_1	$(((320,240),25,1),\ r_{minSur},\ < g_{minSur})$	7000
0.1	q^u_2	$(((320,240),25,1),\ r_{optSur},\ g_{optSur})$	7000
0.13	q^u_3	$(((320,240),50,1),\ r_{optSur},\ g_{optSur})$	11000
...	...	...	...
0.39	$q_{min} = q^u_{16}$	$(((320,240),25,10),\ r_{optSur},\ g_{optSur})$	70000
0.42	q^u_{17}	$(((320,240),25,11),\ r_{optSur},\ g_{optSur})$	77000
...	...	...	...
0.81	q^u_{37}	$(((320,240),75,21),\ r_{optSur},\ g_{optSur})$	315000
0.83	q^u_{38}	$(((320,240),25,24),\ r_{optSur},\ g_{optSur})$	168000
0.84	$q_{opt} \sim q^u_{39}$	$(((320,240),50,23),\ r_{optSur},\ g_{optSur})$	253000
...	...	...	...
1.0	q^u_{45}	$(((640,480),75,25),\ r_{optSur},\ g_{optSur})$	1125000

7 Evaluation

We have used the criterias introduced in [3] to evaluate our specification approach.

7.1 Expressiveness

A network QoS specification technique must provide a sufficient language support to specify the communication requirements of a wide variety of distributed applications. This contains not only the specification of the required resources but also the corresponding adaptation rules.

The presented specification technique enables the specification of network QoS on different system layers with a chosen level of granularity independently of the application. Also the adaptation of the user data flow is adressed by the scalability specification.

7.2 Declarativity

Service user should declaratively specifiy the needed network QoS instead of how this is to be achieved. So, the service users need not to manage complex resource management or control tasks.

The required network QoS is declaratively specified on each system layer. Even the specification of the scalability is confined to an utilitiy and cost function and two thresholds for up- and downscaling respectively. The *how* is not part of the specification at all.

7.3 Independence

A specifcation technique must be independent from any functional specification technique and technological platform. It must also be possible to associate more than one QoS specification to a service user.

It doesen't matter which specific functional specification technique is used in combination with our specification approach, since it is based on a formal mathematical model independently of any functional (specification) language. In addition, it is possible to associate more than one specification to a service user and also to weight the specifications against each other.

7.4 Extensibility

It should be easily possible to extend the QoS specification technique in order to deal with new topics such as security or availability.

It is possible to augment our formalization approach with new types of requirements (*e.g.* security domain) or to extend existing domains (*e.g.* new reliability parameter). This is some effort, however, it is feasible.

7.5 Reuseability

A QoS specification technique should foster reuse. Often, new QoS specifications are just existing ones with minor extensions or refinements. So reuseability would help saving time and effort, especially when specifications become large.

Due to the fact that the basic structure of our technique is based on a metamodel for QoS, reuse is faciliated. Furthermore, it is possible to reuse various parts of the specification, since it is modularly modelled. To specifiy the network QoS requirements of a new audio transmission for a given video transmission on the same platform, you only have to describe the user and application layer and the performance mappings between user - and application - and application - and hardware layer. The resource and midleware layers stay the same. If in addition *e.g.* the reliability requirements of both transmission streams are identical, this component can also be reused.

8 Conclusion and Future Work

We have presented a holistic, comprehensive formalization of network QoS requirements, across layers. QoS requirements are specified on each layer by defining a multi-dimensional QoS domain and QoS scalability. Based on these definitions, we have derived preorders on multi-dimensional QoS domains, and have presented criteria to reduce these domains to manageable subsets, sufficient as a starting point for system design and implementation.

All formalizations so far are based on mathematics. For better usability, we intend to define a formal QoS requirement specification language, with intuitive keywords and structuring capabilities. This language should be powerful enough to host the concepts and criteria we have introduced in this paper.

Another step is to specify designs that satisfy given QoS requirement specifications. In particular, there is need for defining a network QoS system architecture, with QoS functionalities such as QoS provision, QoS control, and QoS management on each abstraction layer. We expect that this requires extensions to existing design languages such as UML or SDL. Finally, implementations are to be generated from design models. In our group, we have a complete development process and tool chain for model-driven development [5]. It is a challenging task to extend them to QoS-aware system development.

Acknowledgment

The work presented in this paper was (partially) carried out in the BelAmI (Bilateral German-Hungarian Research Collaboration on Ambient Intelligence Systems) project, funded by German Federal Ministry of Education and Research (BMBF), Fraunhofer-Gesellschaft and the Ministry for Science, Education, Research and Culture (MWWFK) of Rheinland-Pfalz.

Bibliography

[1] S. Frølund and J. Koistinen. QML: A Language for Quality of Service Specification. Technical Report HPL-98-10, pp. 63., Software Technology Laboratory, Hewlett-Packard Company, 1998.

[2] J. Ø. Aagedal. *Quality of Service Support in Development of Distributed Systems.* PhD thesis, University of Oslo, Oslo, Norway, 2001.

[3] J. Jin and K. Nahrstedt. QoS Specification Languages for Distributed Multimedia Applications: A Survey and Taxonomy. *IEEE MultiMedia*, 11(3):74–87, 2004.

[4] C. Koliver, K. Nahrstedt, J.-M. Farines, J. D. S. Fraga, and S. A. Sandri. Specification, Mapping and Control for QoS Adaptation. *Real-Time Systems*, 23(1-2):143–174, 2002.

[5] T. Kuhn, R. Gotzhein, and C. Webel. Model-Driven-Development with SDL – Process, Tools, and Experiences. In O. Nierstrasz, J. Whittle, D. Harel, and G. Reggio, editors, *Model Driven Engineering Languages and Systems (MoDELS 2006)*, Lecture Notes in Computer Science (LNCS) 4199, pages 83–97. Springer, 2006.

[6] J. Schmitt. *Heterogeneous Network Quality of Service Systems.* Kluwer Academic Publishers, June 2003. ISBN: 07937410X.

[7] D. Schneider, M. Anastasopoulos, J. Bayer, M. Becker, and C. Webel. QoS Specification in Ambient Intelligence Systems. In *Proceedings of ICPS06/SEPS Workshop (SEPS'06)*, Lyon, France, 2006.

[8] R. Vanegas, J. A. Zinky, J. P. Loyall, D. Karr, R. E. Schantz, and D. E. Bakken. QuO's Runtime Support for Quality of Service in Distributed Objects. In *Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing (Middleware'98)*, pages 207–222, The Lake District, UK, 1998.

[9] C. Webel, I. Fliege, A. Geraldy, R. Gotzhein, M. Krämer, and T. Kuhn. Cross-Layer Integration in Ad-Hoc Networks with Enhanced Best-Effort Quality-of-Service Guarantees. In *Proceedings of World Telecommunications Congress (WTC 2006)*, Budapest, Hungary, 2006.