(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2009-0110905
(43) 공개일자 2009년10월23일

(51) Int. Cl.
H04L 12/56 (2006.01) *H04L 12/28* (2006.01)
(21) 출원번호 10-2009-7016889
(22) 출원일자 2008년01월17일
심사청구일자 2009년08월20일
(85) 번역문제출일자 2009년08월13일
(86) 국제출원번호 PCT/US2008/000604
(87) 국제공개번호 WO 2008/088840
국제공개일자 2008년07월24일
(30) 우선권주장
12/009,231 2008년01월16일 미국(US)
60/881,104 2007년01월17일 미국(US)

(71) 출원인
시에라 와이어리스 인코포레이티드
캐나다 브리티쉬 콜롬비아 브이6브이 3에이4 리치몬드 와이어리스 웨이 13811
(72) 발명자
카바노프 리차드 토마스
미국 캘리포니아 92024 엔시니타스 버치뷰 드라이브 736
보스 구스타프 제럴드
캐나다 브리티시 콜롬비아 브이4피 1엔8 써리 25th 애비뉴 14882
(74) 대리인
리앤목특허법인

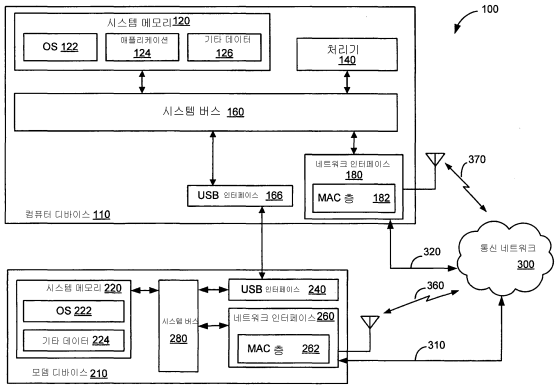
전체 청구항 수 : 총 38 항

(54) 소켓 위의 서비스 응용 프로그래밍 인터페이스의 품질

(57) 요 약

컴퓨터 디바이스상에서 실행되는 QoS-인식 네트워크 애플리케이션을 위한 서비스 품질(QoS) 제어 메커니즘이 개시된다. 이 메커니즘은 네트워크 액세스 디바이스의 MAC 층 QoS 함수를 액세스하기 위한 QoS-인식 애플리케이션을 위해 IP 소켓-기반 QoS 응용 프로그램 인터페이스(API)를 제공한다. 각종 실시예에서, 능동적인 네트워크 인터페이스(들) 위에서 애플리케이션에 의해 전송되는 QoS 데이터를 제어하고 감시하기 위해, 상기 QoS 메커니즘은 네트워크 액세스 디바이스에 소켓(QAoS) 구동기 및/또는 MAC 층 QAoS 핸들러 래퍼 상의 QoS API를 이용한다.

대 표 도 - 도1



특허청구의 범위

청구항 1

하나 이상의 QoS-인식 애플리케이션;

하나 이상의 QoS-지원 네트워크 액세스 디바이스; 및

상기 하나 이상의 QoS-인식 애플리케이션과 상기 하나 이상의 QoS-지원 네트워크 액세스 디바이스의 사이에서 동작하는, QoS 관리 및 보고 통신 프로토콜을 포함하며, 상기 하나 이상의 QoS-인식 애플리케이션이 상기 하나 이상의 QoS-인식 네트워크 액세스 디바이스를 통하여 데이터 통신 스트림의 QoS를 제어하고 관리하는 시스템.

청구항 2

제1항에 있어서, 상기 관리 및 보고 통신 프로토콜은 양방향 접속을 통하여 교환됨을 특징으로 하는 시스템.

청구항 3

제1항에 있어서, 상기 QoS 관리 및 보고 통신 프로토콜 위에서 통신하는 각 QoS-인식 애플리케이션과 QoS-지원 네트워크 액세스 디바이스 쌍 사이에 1대 1의 관계가 있음을 특징으로 하는 시스템.

청구항 4

제1항에 있어서, 상기 네트워크 액세스 디바이스의 QoS-지원 함수가 네트워크 액세스 디바이스에서 구현됨을 특징으로 하는 시스템.

청구항 5

제1항에 있어서, 상기 네트워크 액세스 디바이스의 QoS-지원 함수가 네트워크 액세스 디바이스를 위한 디바이스 구동기에서 구현됨을 특징으로 하는 시스템.

청구항 6

제1항에 있어서, 상기 QoS 관리 및 보고 통신 프로토콜이 잘 알려진 데이터 통신 서비스 상에서 동작함을 특징으로 하는 시스템.

청구항 7

제6항에 있어서, 상기 잘 알려진 데이터 통신 서비스는 인터넷 프로토콜 기반 소켓임을 특징으로 하는 시스템.

청구항 8

제1항에 있어서, 상기 QoS-인식 애플리케이션은 QoS 관리 및 보고 통신 프로토콜을 구현하는 중간 디바이스를 통하여 상기 QoS-지원 네트워크 액세스 디바이스와 통신함을 특징으로 하는 시스템.

청구항 9

제1항에 있어서, 상기 QoS-지원 네트워크 액세스 디바이스는 복수의 네트워크 액세스 디바이스를 위한 중개자인 것을 특징으로 하는 시스템.

청구항 10

제9항에 있어서, 상기 QoS-지원 네트워크 액세스 디바이스는 상기 하나 이상 네트워크 액세스 디바이스 상에서 데이터 트래픽을 경로 설정함을 특징으로 하는 시스템.

청구항 11

제1항에 있어서, 상기 QoS-지원 네트워크 액세스 디바이스는 상기 QoS 관리 및 보고 통신 프로토콜을 사용하여 상기 QoS-인식 애플리케이션에 QoS 상태 정보를 제공하는 것을 특징으로 하는 시스템.

청구항 12

제1항에 있어서, 상기 QoS-지원 네트워크 액세스 디바이스는 데이터 통신 링크를 통하여 QoS-인식 애플리케이션에 원격으로 접속됨을 특징으로 하는 시스템.

청구항 13

제12항에 있어서, 상기 데이터 통신 링크는 유선의 통신 링크인 것을 특징으로 하는 시스템.

청구항 14

제12항에 있어서, 상기 데이터 통신 링크가 무선 통신 링크인 것을 특징으로 하는 시스템.

청구항 15

QoS 관리 및 보고 통신 프로토콜을 사용하여 하나 이상의 QoS-인식 애플리케이션과 하나 이상의 QoS-지원 네트워크 액세스 디바이스 사이에서 데이터를 통신하는 단계; 및

하나 이상의 네트워크 액세스 디바이스를 통하여 하나 이상의 데이터 통신 스트림의 QoS를 제어하고 관리하는 방법.

청구항 16

제15항에 있어서, 상기 관리 및 보고 통신 프로토콜은 양방향 접속을 통하여 교환됨을 특징으로 하는 방법.

청구항 17

제15항에 있어서, 상기 QoS 관리 및 보고 통신 프로토콜 위에서 통신하는 각 QoS-인식 애플리케이션과 QoS-지원 네트워크 액세스 디바이스 쌍 사이에 1대 1의 관계가 있음을 특징으로 하는 방법.

청구항 18

제15항에 있어서, 상기 네트워크 액세스 디바이스의 QoS-지원 함수가 네트워크 액세스 디바이스에서 실현됨을 특징으로 하는 방법.

청구항 19

제15항에 있어서, 상기 네트워크 액세스 디바이스의 QoS-지원 함수가 네트워크 액세스 디바이스를 위한 디바이스 구동기에서 실현됨을 특징으로 하는 방법.

청구항 20

제15항에 있어서, 상기 QoS 관리 및 보고 통신 프로토콜이 잘 알려진 데이터 통신 서비스 상에서 동작함을 특징으로 하는 방법.

청구항 21

제20항에 있어서, 상기 잘 알려진 데이터 통신 서비스는 인터넷 프로토콜 기반 소켓임을 특징으로 하는 방법.

청구항 22

제15항에 있어서, 상기 QoS-인식 애플리케이션은 QoS 관리 및 보고 통신 프로토콜을 구현하는 중간 디바이스를 통하여 상기 QoS-지원 네트워크 액세스 디바이스와 통신함을 특징으로 하는 방법.

청구항 23

제15항에 있어서, 상기 QoS-지원 네트워크 액세스 디바이스는 복수의 네트워크 액세스 디바이스를 위한 중개자인 것을 특징으로 하는 방법.

청구항 24

제23항에 있어서, 상기 QoS 네트워크 액세스 디바이스는 상기 하나 이상 네트워크 액세스 디바이스 상에서 데이터 트래픽을 경로 설정함을 특징으로 하는 방법.

청구항 25

제15항에 있어서, 상기 QoS-지원 네트워크 액세스 디바이스는 상기 QoS 관리 및 보고 통신 프로토콜을 사용하여 상기 QoS-인식 애플리케이션에 QoS 상태 정보를 제공하는 것을 특징으로 하는 방법.

청구항 26

제15항에 있어서, 상기 QoS-지원 네트워크 액세스 디바이스는 데이터 통신 링크를 통하여 QoS-인식 애플리케이션에 원격으로 접속됨을 특징으로 하는 방법.

청구항 27

제26항에 있어서, 상기 데이터 통신 링크는 유선의 통신 링크인 것을 특징으로 하는 방법.

청구항 28

제26항에 있어서, 상기 데이터 통신 링크가 무선 통신 링크인 것을 특징으로 하는 방법.

청구항 29

QoS-인식 애플리케이션과 QoS-지원 네트워크 액세스 디바이스 사이에서 양방향 접속을 설정하는 단계; 및

상기 설정된 양방향 접속을 통하여 QoS-관련 데이터를 통신하는 단계를 포함하는 방법.

청구항 30

제29항에 있어서, 상기 양방향 접속이 IP 소켓인 것을 특징으로 하는 방법.

청구항 31

제29항에 있어서, 상기 설정 단계는 QoS-인식 애플리케이션과 QoS-지원 네트워크 액세스 디바이스의 MAC 층 사이에서 양방향 접속을 설정하는 단계를 포함하는 것을 특징으로 하는 방법.

청구항 32

제29항에 있어서, 상기 통신 단계는 QoS 관리 및 보고 통신 프로토콜을 통하여 QoS-관련 데이터를 통신하는 단계를 포함하는 것을 특징으로 하는 방법.

청구항 33

제29항에 있어서, 상기 통신 단계는 QoS-관련 데이터를 구비한 하나 이상의 QoS 메시지를 송신하는 단계를 포함하는 것을 특징으로 하는 방법.

청구항 34

제29항에 있어서, 상기 QoS-지원 네트워크 액세스 디바이스로 양방향 접속을 설정하기 위한 하나 이상의 QoS API를 제공하는 단계를 더 포함하고 있는 것을 특징으로 하는 방법.

청구항 35

제29항에 있어서, 상기 QoS-지원 네트워크 액세스 디바이스에 QoS-관련 데이터를 통신하기 위한 하나 이상의 QoS API를 제공하는 단계를 더 포함하고 있는 것을 특징으로 하는 방법.

청구항 36

제29항에 있어서, QoS 애플리케이션과 통신하기 위한 MAC 층 QoS API 핸들러를 제공하는 단계를 더 포함하고 있는 것을 특징으로 하는 방법.

청구항 37

제36항에 있어서, 상기 MAC 층 QoS API 핸들러가 상기 네트워크 액세스 디바이스를 위한 디바이스 구동기로 구현됨을 특징으로 하는 방법.

청구항 38

제36항에 있어서, 상기 MAC 층 QoS API 핸들러가 상기 네트워크 액세스 디바이스의 MAC 층에 구현됨을 특징으로 하는 방법.

명 세 서

기 술 분 야

<1> 관련 출원들의 상호 참조

<2> 본 출원은 2007년 1월 17일 출원된 미국 가출원의 출원 번호 제60/881,104호의 우선권 주장 출원이며, 여기에 참조로써 전체적으로 통합된다.

<3> 기술분야

<4> 본 출원의 개시는 통신 네트워크 분야에 관한 것으로, 보다 상세하게는 네트워크 애플리케이션의 서비스 품질(QoS) 지원에 관한 것이다.

배 경 기 술

<5> 네트워크 가능 컴퓨팅 시스템 및 네트워크가 유비쿼터스하고 더욱 다양하게 됨에 따라, 이들 네트워크를 이용하는 애플리케이션의 데이터 통신 요건도 역시 유비쿼터스하고 더욱 다양하게 되었다. 컴퓨팅 디바이스의 확산과 더불어, 이들 통신 서비스를 위해 필요한 서비스 품질(QoS)을 제공할 필요가 증대하고 있다. QoS는 일반적으로 각종 네트워크에 흐르고 있는 데이터를 위해 네트워크 애플리케이션에 보증된 레벨의 성능을 유지하는 것을 허락하는 자원 예약 제어 메커니즘에 관한 것이다. QoS 메커니즘을 사용하면서, 네트워크 액세스 디바이스, 이를테면 모뎀, 라우터, 스위치, 및 네트워크 서버는 다른 애플리케이션, 사용자 또는 데이터 흐름에 또는 어떤 레벨의 성능을 보증하기 위하여 다른 우선권을 제공할 수 있다.

<6> 특정의 QoS를 필요로 할 수 있는 애플리케이션 네트워크 트래픽 유형은 스트리밍 멀티미디어, IP 전화, 게임, 및 기타 다른 것들을 포함한다. 각 유형의 네트워크 트래픽은 다른 유형의 데이터 배달 서비스를 필요로 한다. 예를 들면, 스트리밍 멀티미디어는 전형적으로 신뢰성있는 데이터 배달의 높은 보증된 처리율과 로우 레벨의 지연 가변성을 필요로 한다. 반면에, IP 전화는 전형적으로 지연과 지터(jitter)에 대해 엄격한 제한을 요구하지만, 데이터 손실에 대해서는 훨씬 관대하다. 그러므로, 네트워크가 각 애플리케이션에 다른 QoS를 제공하여, 사용자의 기대치를 만족시키고 그 특정 애플리케이션에 대해 좋은 사용자 경험을 제공하는 것이 바람직하다.

<7> QoS-인식(aware) 애플리케이션에 대해 그것의 네트워크 트래픽에 필요한 QoS를 확립하고 유지하는 하나의 공지된 접근법은 애플리케이션이 실행하는 컴퓨팅 디바이스의 QoS 애플리케이션 프로그래밍 인터페이스(API)이다. 전형적으로, 컴퓨팅 디바이스는 운영 체계(OS) 또는 플랫폼 특화의 라이브러리 또는 서비스의 일부로서 QoS API들을 제공한다. 그러므로, 그런 QoS API들은 본질적으로 OS 또는 플랫폼 에 특화된다. 이에 따라서, 다양한 OS, 플랫폼 및 네트워크에 걸쳐 QoS-지원을 가지는 네트워크 애플리케이션을 설계하는 것은 어렵다. 전형적으로 그렇게 하기 위한 시도는 API들이 매우 복잡해지게 하거나, 또는 너무 추상적이어서 애플리케이션에 QoS 레벨의 어떤 의미가 있는 사양도 제공할 수 없는 QoS 조잡성(coarseness)과 투명성의 레벨을 API들이 가지게 한다.

<8> 이 추상성은 MAC 층 함수의 QoS API 없는 상세한 제어 모니터링을 변함없이 가져온다. ISO OSI 구조에서, 매체 접근 제어(MAC) 함수는 네트워크에 액세스하는 디바이스의 능력을 지배하고, 데이터 통신 서비스의 총체적인 QoS에 중요한 영향을 끼친다. 또한, MAC 함수는 간섭의 레벨과 신호의 강도와 같은, 따라서, 네트워크 트래픽의 QoS의 레벨을 감시할 때 중대한, 네트워크의 동적 조건을 측정한다. 그러므로, 네트워크 애플리케이션을 위해 QoS의 유효하고 성공적인 지원을 확실하게 하기 위해 MAC 함수가 QoS 서비스 솔루션의 부분인 것은 바람직하다.

발명의 상세한 설명

<9> 컴퓨터 디바이스로 IP 소켓을 경유하여 네트워크 디바이스의 QoS 함수에 액세스하기 위해 실행하고 있는 QoS-인식 애플리케이션을 위해 QoS API를 제공하는 QoS 제어 메커니즘이 개시된다. 일 예의 실시예에서, 비(non)-QoS-인식 운영 체계와 연관된 API는 QoS APIs로 교체될 수 있다. 애플리케이션을 위하여 개선된 QoS-지원을 달

성하기 위해, QoS API는 네트워크 디바이스의 QoS 함수와 협력할 수 있다. 하나의 실시예에서, QoS-지원은 물리적 네트워크 디바이스의 펌웨어에서 및/또는 디바이스를 위한 디바이스 구동기(들)에서 구현될 수 있다. 다른 실시예에서, 원하는 QoS 서비스 수준을 확실하게 하기 위해 다중 네트워크 기기들에 걸쳐 통신을 관리하고 다중 네트워크를 가로질러 데이터 통신을 조정하는 상급의 디바이스 구동기의 구현이 제공될 수 있다.

<10> 일 예의 실시예에서, 신규 QoS 제어 메커니즘은 활동적인 네트워크 인터페이스 또는 인터페이스들 상에서 애플리케이션에 의해 전송되고 수신하고 있는 데이터의 QoS를 제어하고 감시하기 위해 소켓 상 QoS API(QoS API over Socket; QAoS) 핸들러 및/또는 MAC 층 QAoS 핸들러 래퍼를 이용한다. 이 메커니즘은 컴퓨터 디바이스의 시스템 버스에 직접 접속되거나, 또는 외부 인터페이스를 통해 간접적으로 접속된 네트워크 액세스 디바이스에 대해 지원을 허용한다. 이 지원은 주로 OS 및 플랫폼으로부터 독립된 방법으로 제공된다. 애플리케이션에서 OS-특화 또는 플랫폼-특화 QoS API의 지원 또는 사용은 필요하지 않을 수 있다. 또한, 개시된 QoS 제어 메커니즘은 OS 또는 플랫폼에 변경을 필요로 하지 않고 새로운 네트워크와 MAC 층 QoS 계획을 위해 장래의 개정판에 부가할 규모 확장성(scalability)을 제공한다. 개시된 QoS 제어 메커니즘의 다른 이익은 당해 기술에 숙련된 당업자들에게는 명백할 것이다.

<11> 일 예의 실시예에서, QoS 제어를 위한 시스템은 하나 이상의 QoS-인식 애플리케이션과 하나 이상의 QoS-지원 네트워크 액세스 디바이스를 포함한다. 시스템은 하나 이상의 QoS-인식 애플리케이션과 하나 이상의 QoS-지원 네트워크 액세스 디바이스 사이에서 동작하는 QoS 관리 및 보고 통신 프로토콜을 더 포함한다. 이 프로토콜은 양방향 접속, 이를테면 인터넷 프로토콜(IP) 소켓을 통하여 교환된다. QoS-인식 애플리케이션은 네트워크 액세스 디바이스를 통하여 데이터 통신 스트림의 QoS를 제어하고 관리한다.

<12> 다른 예의 실시예에서, QoS 제어의 방법은 하나 이상의 QoS-인식 애플리케이션과 하나 이상의 QoS-지원 네트워크 액세스 디바이스 사이에서 QoS 관리 및 보고 통신 프로토콜을 사용하여 데이터를 통신하는 단계를 포함한다. 이 방법은 네트워크 액세스 디바이스를 통하여 하나 이상의 데이터 통신 스트림을 위해 QoS를 제어하고 관리하는 단계를 더 포함한다. 관리 및 보고 통신 프로토콜은 양방향 접속, 이를테면 IP 소켓을 통하여 교환된다.

<13> 다른 예의 실시예에서, QoS 제어의 방법은 QoS-인식 애플리케이션과 QoS-지원 네트워크 액세스 디바이스 사이의 양방향 접속을 확립하고, 확립된 양방향 접속을 통하여 QoS-관련 데이터를 통신하는 단계를 포함한다. 양방향 접속은 IP 소켓일 수 있다. 이 접속은 QoS-지원 네트워크 액세스 디바이스의 QoS-인식 애플리케이션과 MAC 층 사이에서 확립될 수 있다. QoS-관련 데이터는 QoS 관리 및 보고 통신 프로토콜을 사용하여 통신될 수 있다.

실 시 예

<19> 당해 기술의 그 통상의 숙련자는 QoS 제어 시스템과 방법의 다음 설명이 실예가 될 뿐이고 어쨌든 제한할 의도는 없는 것을 알 것이다. 다른 실시예는 이 개시의 이점을 가지고 그런 숙련자들에게 쉽게 제안될 것이다. 지금 첨부도면을 참조하여 실시예의 구현에 대한 설명이 상세하게 이루어질 것이다. 동일하거나 유사한 항목을 참조하기 위해 동일한 참조 표시가 도면과 다음 설명 전체에 걸쳐서 가능한 범위까지 사용될 것이다.

<20> 명료함을 위해, QoS 제어 메커니즘의 구현의 루틴 특징 모두가 도시되고 기술된 것은 아니다. 개발자의 특정의 목표, 이를테면 애플리케이션-관련, 시스템-관련, 네트워크-관련, 및 비즈니스-관련 제약의 준수를 달성하기 위해, 네트워크 액세스 메카니즘의 실제 구현 개발시, 다수의 구현-특유한 결정이 만들어져야 하고, 이들 특정의 목표는 구현마다 그리고 개발자마다 다를 것이다라는 것을 이해할 수 있을 것이다. 더욱이, 개발 노력이 복잡하고 시간이 걸릴 수도 있으나 그럼에도 불구하고 이 개시의 이점을 가지고 원격 통신 네트워크 분야의 그 통상의 기술을 가진 자들에게는 엔지니어링의 루틴한 작업 일일 것이다고 이해될 것이다.

<21> 이 개시에 따라, 여기에 기술된 구성요소, 공정 단계 및/또는 데이터 구조는 운영 체계, 컴퓨팅 플랫폼, 네트워크 디바이스, 컴퓨터 프로그램 및/또는 범용 기계의 각종 유형을 사용하면서 실행될 수 있다. 또한, 당해 기술의 그 통상의 기술자는 덜 일반적인 성질, 이를테면 물리적으로 접속되는 디바이스, 필드 프로그래머블 게이트 어레이(FPGAs), 주문형 반도체(ASIC), 또는 그 종류의 다른 것의 디바이스가 또한 여기에 개시된 발명 개념의 범위와 정신에서 이탈하지 않고 사용될 수 있다고 인정될 것이다. 일련의 공정 단계를 포함하고 있는 방법이 컴퓨터 또는 기계로 이행되는지와 그 공정 단계가 기계로 가독가능한 일련의 명령어로 저장될 수 있을 때, 그들은 유형의 매체에서 저장될 수 있다.

<22> 도 1의 블록도는 여기에 기술된 원리에 의거하여 하나 이상의 통신 네트워크 상에서 데이터 통신을 수행하는 능력을 구비한 컴퓨터 시스템의 일 예의 실시예를 도시한다. 컴퓨터 시스템(100)은 개인용 컴퓨터, 랩톱 컴퓨터, 태블릿 컴퓨터, 노트북 컴퓨터, 울트라-모바일 개인용 컴퓨터, 서버, 이동 전화, 개인 휴대 정보 단말기, 멀티

미디어 디바이스, 이를테면 오디오 플레이어, 비디오 재생기, 게임 기계 콘솔, 디지털 카메라, 비디오 카메라와 네트워크(300) 상에서 데이터 통신을 수행할 수 있는 다른 어느 컴퓨터 제어 디바이스를 포함할 수 있으나 이에 제한되는 것은 아니다.

<23> 통신 네트워크(300)는 유선의 통신네트워크(310, 320) 및 무선 통신네트워크 (360, 370)를 포함할 수 있으나 이에 제한되는 것은 아니다. 예를 들면, 통신 네트워크(300)는 이더넷(IEEE 802.3), IEEE 802.11 또는 다른 현재이거나 장래의 LAN 시스템을 이용하는 무선 근거리 네트워크(WLAN), 3GPP(예를 들면 UMTS, HSDPA, LTE), 3GPP2(예를 들면 CDMA, EVDO), 또는 다른 현재이거나 장래의 WWAN 시스템을 이용하는 무선 광역 통신망(WWAN), WiMAX(IEEE 802.16) 또는 다른 현재이거나 장래의 WMAN 시스템을 이용하는 무선 대도시 통신망(WMAN)시스템, IEEE 802.15, 블루투스 그리고 다른 WPAN 시스템을 이용하는 무선 사설망(WPAN)을 포함할 수 있으나 이에 제한되는 것은 아니다.

<24> 일 예의 실시예에서, 컴퓨터 시스템(100)은 Intel® Dual-Core™ 또는 Pentium® 프로세서, AMD Turion™ 64 프로세서 또는 다른 유형의 CPU와 같은 처리 디바이스(140)를 포함하는 범용 컴퓨터 디바이스(110)를 포함할 수 있다. 이 컴퓨터 시스템(100)은 디바이스(110)의 구성요소들을 상호 접속하는 시스템 버스(160)를 더 포함한다. 디바이스(110)는 운영체계(OS)(122)를 저장하는 시스템 메모리(120)를 더 포함한다. 시스템 메모리(120)는 또한 처리 디바이스(140)에서 실행되는 하나 이상의 QoS-인식 애플리케이션(124)와 다른 유형의 데이터(126)를 저장할 수 있다. 시스템 메모리는 랜덤 액세스 메모리(램; RAM), 읽기 전용 메모리(롬; ROM), 프로그램 가능 ROM(PROM), 소거 가능 PROM(EPROM), 플래시-EPROM과 다른 유형의 동적, 휘발성과 비휘발성 정보 기억 매체를 포함할 수 있다. 디바이스(110)는 시스템 버스(160)를 통하여 컴퓨터 디바이스(110)에 직접 접속된 하나 이상의 유선 및/또는 무선 네트워킹 인터페이스 디바이스(180), 및 하나 이상의 외부 디바이스 인터페이스, 이를테면 USB 시리얼 버스(166), 직렬 접속 포트와 IEEE 1394를 더 포함한다.

<25> 일 예의 실시예에서, 외부 디바이스 인터페이스, 이를테면 인터페이스(166)는 컴퓨터 시스템(100)을 대신하여 네트워크 상에서 데이터 통신을 수행 가능한 하나 이상의 컴퓨터 디바이스에 접속될 수 있다. 그런 외부 디바이스는 모뎀 디바이스(210), 이를테면 다이얼 업 모뎀, 케이블 모뎀, DSL 모델 또는 다른 네트워크 액세스 디바이스, 그런 라우터, 스위치, 무선 액세스 포인트 등일 수 있다. 모뎀 디바이스(210)는 제어와 네트워킹 트래픽 데이터를 교환하는 컴퓨터 디바이스(110)에 대한 접속을 제공하는 상응하는 외부 디바이스 인터페이스, 이를테면 USB 인터페이스(240)와 하나 이상의 무선 및/또는 유선 네트워킹 인터페이스 디바이스(260)를 포함한다. 그런 모뎀 디바이스(210)는 또한 독립형의 OS(222)와 다른 데이터(224)가 있는 내부 시스템 메모리(220), 뿐만 아니라 모뎀 디바이스(210)의 복잡도에 따라 내부 시스템 버스(280)를 포함할 수 있다.

<26> QoS-인식 애플리케이션(124)이 네트워크(300) 위에서 데이터 통신을 위한 원하는 QoS를 달성하게 하기 위해서는, 컴퓨터 디바이스(110)가 통신네트워크(300)의 QoS 능력에 대한 인터페이스 없이 QoS의 원하는 레벨을 달성하고 보증할 수 있는 최고의 노력보다 더 높은 QoS의 수준을 애플리케이션(124)이 요구한다는 가정하에, 하나의 인터페이스가 애플리케이션(124)과 통신네트워크(300)의 QoS 능력에 액세스하는 것에 대해 책임이 있는 네트워크 인터페이스 디바이스(180, 260)의 MAC 층(182, 262)의 요소 사이에 존재하여야함은 당해 기술에 숙련된 기술자에게는 이해될 것이다. 따라서, QoS-인식 애플리케이션(124)과 네트워크 인터페이스 디바이스(180, 260)의 QoS 함수 사이에 고유의 소켓-기반(socket-based) 또는 다른 유형의 양방향 링크에 대한 요구가 있다.

<27> 일 예의 실시예에서, 신규 QoS 제어 메커니즘은 네트워크 인터페이스 디바이스(180, 260)의 MAC 층 QoS 함수에 액세스하기 위해 컴퓨터 디바이스(110)에서 실행하고 있는 QoS-인식 애플리케이션(124)에 QoS API를 제공한다. 일 예의 실시예에서, QoS 제어 메커니즘은 컴퓨터 디바이스(110)의 시스템 버스(160)에 직접 접속되거나, 또는 외부 인터페이스, 이를테면 USB 인터페이스(166)를 통해 간접적으로 접속된 네트워크 액세스 디바이스를 위해 지원을 제공한다. 이 지원은 주로 OS-그리고 플랫폼에 의존하지 않은 방법으로 제공된다. 애플리케이션(124)의 OS-또는 플랫폼에 특유한 QoS API 지원이 필요하지 않을 수 있음은 당해 기술에 숙련자들에게 이해될 것이다.

<28> 일 예의 실시예에서, 네트워크 액세스 메카니즘은 QoS API 메시지를 전송하기 위해 사용되는 쉽게 그리고 잘 인식될 수 있는 데이터 통신용 인터페이스, 이를테면 소켓-기반, IP-기반 또는 기타 다른 것들을 포함할 수 있다. 이들 QoS API 메시지는 애플리케이션(124)과 직접 또는 간접적으로 접속된 네트워크 인터페이스 디바이스(180, 260)의 MAC 층 사이에서 전송될 수 있다. 그런 데이터 인터페이스의 존재가 애플리케이션과 네트워크 디바이스의 MAC 층 사이에서 어떤 네트워크 트래픽 데이터를 전달하거나, 수신하기 위해 전형적으로 사용되므로, QAoS(QoS API over Socket) 메시징을 전송하는 데 필요한 데이터 스택은 전형적으로 그런 컴퓨팅 디바이스에 그

리고 네트워크 인터페이스 디바이스(180, 260)에 존재할 것이다.

<29> 네트워크 액세스 메카니즘의 일 예의 실시예에서, 네트워크 인터페이스 디바이스(180, 260)의 MAC 층의 QAoS 능력은 MAC 층 QAoS 핸들러(MQA)에 구현될 수 있다. MQA는 네트워킹 인터페이스 디바이스의 실제 MAC 층 또는 하나 이상의 상응하고 있는 네트워크 인터페이스 디바이스를 위한 하나 이상의 MAC 층 주위의 래퍼로서 생성될 수 있다. MQA가 MQA를 천성적으로(natively) 지원할 수 있거나, 지원할 수 없는 MAC 층 주변의 래퍼로서 존재하는 경우, MQA는 OS(122)의 일부로서 또는 다른 방법으로, 네트워크 인터페이스 디바이스 내에, 컴퓨터 디바이스(110)와 인터페이스하기 위한 하나 이상의 네트워크 인터페이스 디바이스용 디바이스 또는 중간 디바이스 구동기 내에 존재할 수 있다. 후자의 경우는 컴퓨터 시스템(100)이 둘 이상의 네트워크를 동시에 지원할 때 사용될 수 있고, 데이터 통신이 시간에 따라서 다른 네트워크(320, 370)의 사이에 핸드-오프하고 있을 때 애플리케이션(124)과 MQA 사이에서 QAoS 소켓을 유지하는 것은 바람직하다.

<30> 도 3은 애플리케이션(510)이 실행중인 컴퓨터 디바이스의 OS 및 플랫폼의 전송 프로토콜 층(들)(540) 및 네트워크 프로토콜 층(들)(545)을 통하여 QoS-인식 애플리케이션(510)과 네트워킹 인터페이스 디바이스(592, 594,(596, 598))의 MAC 층(573, 565, 577, 569) QoS 관리 요소 사이의 소켓(522, 524, 527), 이를테면 TCP/IP 또는 UDP/IP 소켓 상에서 터널을 파게 되는 QoS API 메시징 프로토콜을 통하여 QoS-인식 애플리케이션(510)에 의해 원해지는 QoS의 레벨을 달성하는 신규 QoS 제어 메커니즘의 일 예의 실시예를 도시한다. 개시된 소켓상 QoS API(QAoS) 메커니즘은 그런 QoS 관리 요소에 인터페이스하기 위해 QAoS 메시징을 위한 메커니즘이 있다라는 가정하에, 주로 네트워킹 인터페이스 디바이스의 MAC 층(573, 565, 577, 569) QoS 관리 요소에 인터페이스하기 위해 OS-독립 및 플랫폼-독립 방법을 제공한다.

<31> 일 예의 실시예에서, QAoS 가능한 애플리케이션(510)은 QAoS 핸들러(520)를 포함할 수 있다. 애플리케이션(510)의 QoS 요건을 만족시키기 위한 시도로, 핸들러(520)는 비-QAoS 데이터의 통신을 위해 사용되는 동일하거나 유사한 소켓 인터페이스 상에서 QAoS 소켓(522, 524, 527)을 시작하고, 유지하고, 제거하기 위해 구성된다. 그러므로, 그 애플리케이션(510)이 QAoS 가능한 네트워킹 인터페이스 디바이스를 가로질러 비-QAoS 데이터 통신을 벌써 지원하는 OS 및 플랫폼을 위해 QAoS 인터페이스의 사용을 지원하는 애플리케이션(510)에 대해 추가 요구되는 개발은 거의 없다. 또한, 다른 OS들 및 플랫폼들을 가로지르는 QAoS를 위한 이식 지원(porting support)을 위한 노력의 레벨은 동일한 세트의 다른 OS들 및 플랫폼들을 가로지르는 비-QAoS 데이터 통신을 위한 이식 지원을 위한 노력으로 주로 제한된다.

<32> 하나의 실시예에서, MAC QAoS API 핸들러(MQA)는 애플리케이션(510)으로부터 어떤 QAoS 소켓(522, 524, 527)을 종료하기 위해 제공된다. MQA(573, 565, 569, 577)는 QAoS 소켓의 설정, 유지 보수와 종료의 서버 타입 관리를 용이하게 한다. MQA(573, 565, 569, 577)는 애플리케이션(510)에 의해 수신한 QAoS 소켓 메시징 요구를 기본의 네트워크와 네트워크 인터페이스 디바이스(592, 594, 596, 598)의 MAC 층 프로토콜 메시징으로의 번역(ranslation)과 발송(forwarding)을 가능하게 한다. MQA(573, 565, 569, 577)는 어떤 관련 MAC 층 QoS 상태도 및 이벤트 통지를 번역하고 QAoS 소켓(522, 524, 527)을 통하여 애플리케이션(510)에 보고한다. MQA(573, 565, 569, 577)는 애플리케이션(510)부터 MAC 층까지 이전에 전송된 어떤 요구에 대하여 MAC 층(573, 565, 577, 569) QoS 관리 요소로부터 수신받는 어떤 응답도 번역하고, 전송한다. MQA(573, 565, 569, 577)은 당해 기술에 숙련된 자들에게 공지된 다른 함수를 제공하기 위해 또한 구성될 수 있다.

<33> 일 예의 실시예에서, MQA(573, 577)는 MAC 층(572, 576)의 안에 본질적으로(natively) 존재할 수 있다. 그러나, 다른 예의 실시예에서, 만일 네트워킹 인터페이스 디바이스의 MAC 층이 MQA를 본질적으로 포함하지 않으면, MAC QAoS 핸들러(MQA) 래퍼(565, 569)가 애플리케이션(510)이 실행하고 있는 컴퓨터 디바이스의 네트워킹 스택(545)과 네트워크 인터페이스 디바이스 (594, 598)의 MAC 층(574, 578) 사이에 어디인가에 존재하여, 인터페이스(565b, 569b)가 MAC 층(574, 578)의 QoS 관리 요소에 QAoS 메시징을 위해 있을 것이다. 당해 기술의 숙련자들은 위의 예가 제한적이지 않고 MQA가 컴퓨터 시스템의 다른 형식과 다른 위치에 존재할 수 있음을 인식할 수 있다.

<34> 일 예의 실시예에서, MQA 래퍼(565, 569)는 네트워킹 인터페이스 디바이스 (594, 598)를 위해 제공된 디바이스 구동기(564, 568)에 위치할 수 있으므로, 비-QAoS 데이터 통신을 위해 필요한 구동기가 애플리케이션(510)이 실행하고 있는 컴퓨터 디바이스의 OS와 플랫폼에 존재하고 설치되는 한, 디바이스는 디바이스가 마치 OS, 플랫폼과 애플리케이션(510)에 MQA 지원을 본질적으로 제공하는 것처럼 나타난다. 그러나, MQA가 네트워킹 인터페이스 디바이스의 디바이스 구동기보다 높은 프로토콜 계층에 위치할 수 있으므로, 애플리케이션(510)이 실행하고 있는 컴퓨터 디바이스의 OS와 플랫폼에서 더 많은 QAoS 종속성을 생성할 수 있다. 이는 덜 원하는 것이지만, 각

종 실시예의 추가 설명으로부터 명백하게 될 이 접근법에는 다른 이점이 있다.

<35> 하나의 컴퓨터 시스템이 동시에 몇 개의 네트워크(예를 들면, 유선과 무선이다)로 연결성을 지원하는 경우에는, 애플리케이션(510)은 MQA로 그 QAoS 소켓을 유지하기 위해 구성될 수 있으며, 한편 데이터 통신은 시간에 따라 핸드오프되고 및/또는 다른 둘 이상의 네트워크 인터페이스-디바이스(596, 598)를 경유하여 다른 네트워크의 사이에 동시에 동적으로 또는 정적으로 분산된다. 그 때문에, 다중 MAC QAoS 핸들러 중재인(MMQAA)(548)은 애플리케이션(510)과 둘 이상의 MQA들 및/또는 MQA 래퍼들(577, 569) 사이의 단일 셋트의 QAoS 소켓(527, 517)을 종료하고 유지하기 위해 제공될 수 있기 때문에, 만일 다중 네트워크 접속을 가로지르는 이 무결절성 QoS 동작이 애플리케이션(510)에 의해 요망되면, 네트워크와 상응하는 네트워크 인터페이스 디바이스(596, 598) 사이의 핸드오프에 기인하는 QoS 관련 동작, 또는 동일한 네트워크를 가로지르는 다이내믹 또는 정적인 데이터 분배가 QAoS 소켓(527, 517)을 이용하는 애플리케이션(510)에 투명하다.

<36> 일 예의 실시예에서, MMQAA(548)는 중간 디바이스, 이를테면 중간 구동기(547) 내에서 포함될 수 있으며, 이는 네트워킹 인터페이스 디바이스(596, 598)로 직접 또는 간접적으로 연관될 수 있다. 이처럼 MMQAA(548)는 OS 및 하부에 있는 컴퓨팅 플랫폼으로부터 비교적 독립적이다. 더욱이, MMQAA(548)는 QAoS 소켓(527, 517)과 QAoS 소켓(527, 517)상에서 수행되는 전송된 동작들의 중재를 하도록 구성될 수 있어, 네트워크 인터페이스 디바이스들을 가로질러 비-QAoS 데이터 통신 중재에 요구되는 구동기들이 애플리케이션(510)이 실행하고 있는 컴퓨터 디바이스의 OS와 플랫폼에서 설치되고 존재하는 한 중간 구동기(547)가 존재하여야 한다.

<37> 다른 예의 실시예에서, MMQAA(548)는 애플리케이션(510)이 실행하고 있는 컴퓨터 디바이스의 OS 나 플랫폼 특화 부분 내에 포함될 수 있다. 이 경우, MMQAA(548)는 컴퓨터 시스템에 접속된 어떤 가용 네트워킹 인터페이스 디바이스를 가로질러 애플리케이션(510)과의 무결절성 및 투명한 QAoS 소켓들 및 동작을 유지하도록 구현될 수 있으며, 이를 위해 MMQAA(548)는 네트워크 인터페이스 디바이스의 MQA와 동일한 컴퓨터 시스템에 접속된 하나 이상의 네트워크 인터페이스 디바이스의 MQA(들) 사이에서 QAoS 동작의 중재를 하는 능력을 갖고 있다. 당해 기술에 숙련된 자는 상기 예가 비제한적이고 MMQAA가 컴퓨터 시스템의 다른 형식으로 및 다른 위치에 존재할 수 있음을 알 수 있다.

<38> OS 또는 플랫폼이 기존의 OS 또는 플랫폼 특화 QoS API에 반대하는 OS-특화 또는 플랫폼-특화 QoS API(530)를 이미 이용하는, 레가시(legacy) 애플리케이션(510)에 대비하기 위하여, 일 예의 실시예에 의거하여 QAoS 심(shim)(535)이 제공될 수 있다. QAoS 심(535)은 어떤 접속된 MQA-가능(capable) 네트워크 인터페이스 디바이스(592, 594, 596, 598)의 QAoS 메시징 소켓(512, 514, 517) 상에서 OS 또는 플랫폼 특화의 QoS API(530)와 MQA들(573, 565, 577, 569) 사이에 어떤 필요한 번역(531)도 수행할 수 있다. 예를 들면, QoS 요청과 QoS API(530)에 의해 제공된 응답의 포맷은 사용중인 MQA에서 지원된 QoS 요구와 응답 메시지에 대응될 수 없다(1대 1 맵핑될 수 없다) (즉, MQA에서 지원된 QoS 요구와 응답 메시지들은 다른 QoS 제어와 상태 메시징 프로토콜 언어/API들을 의미할 수 있다). 이 경우, QAoS 심(535)은 사용중인 OS/플랫폼과 MQA 사이에서 QoS API 언어의 번역을 수행할 수 있다. 다른 예에서, OS/플랫폼-특화 QoS API(530)는 단지 하이-레벨 (가령 서비스 계층) QoS API를 QoS-인식 애플리케이션(510)에 제공할 수 있고, QAoS 심(535)은 상위 레벨 QoS API들 사이에서 사용중인 MQA에 의해 지원된 하위 레벨(예를 들면 네트워크 계층) QoS API들로 번역할 수 있다. 또다른 예에서, QoS 서비스 흐름, 분류 등, OS/플랫폼-특화 QoS API(530)를 통하여 애플리케이션에 의해 명시된 속성들은 사용중인 대응 네트워크 구성 인터페이스 디바이스(들)의 MQA에 의해 지원되는 동일한 세분성(granularity), 범위(ranges), 열거(enumerations), 등을 지원할 수 없다. 그러므로, QAoS 심(535)은 이들 API 속성 차이들을 번역할 것이다. 다른 예에서, OS/플랫폼-특화 QoS API는 OS/플랫폼(예를 들면 802.3과 802.11)의 발단(inseption) 시점에서 산업상 이용할 수 있는 라디오 액세스 기술의 QoS 능력을 지정하기 위해 오로지 디자인될 수 있다. 이 경우, 규모 확장성을 위해, QAoS 심(535)은 새로운 라디오 액세스 기술(예를 들면, 802.16m)로 지원된 새로운 QoS 능력을 위해 필요한 번역을 제공할 수 있는데, 이는 OS/플랫폼-특화 QoS API를 업그레이드할 필요가 없이도 이용될 수 있기 때문이다. 당해 기술에 숙련된 자들은 레가시 애플리케이션 또는 컴퓨터 시스템을 구비한 개시된 QoS 제어 메커니즘의 실시예를 통합하는 다른 방법이 있음을 인식할 수 있다.

<39> 당해 기술에 숙련된 자들은 MQA-가능 네트워크 디바이스가 하나의 IP 주소를 가질 때, QoS-인식 애플리케이션에 의해 이 IP 주소에 송신된 메시지가 QoS-인식 애플리케이션이 실행하고 있는 컴퓨터 디바이스의 OS에 의해 순환(looped back)될 수 있음을 인식할 수 있다. 이 문제를 피하기 위해, MQA-가능 네트워크 디바이스를 위해 수용하는 OS 경로 테이블에 추가 기입(entry)이 이루어질 수 있다. MQA는 그 할당된 IP 주소에서 착신 데이터 트래픽 필터링을 감시할 수 있다. 만일 경합 대상이 있고 QAoS 메시지가 아니면, OS로 순환될 것이다(이는 역방향 호환성을 유지한다). 만일 경합 대상이 있고 유효한 QAoS 메시지라면, MQA에 의해 처리된다. QoS-인식 애플리

케이션은 네트워크 디바이스가 IP 주소를 얻었을 때를 표시하는 OS 콜백을 감시하거나 등록할 수 있다. 이는 새로운 네트워크 디바이스의 주소로 설정된 착신지 주소를 갖는 QAoS 패킷을 생성하여, MQA-가능 네트워크 디바이스를 가지고 QAoS 세션을 시작하기 위한 애플리케이션에 대한 트리거일 것이다. 고유 포트 번호는 보통의 TCP 세션 방법에 의해 결정될 수 있다.

<40> 따라서, 할당된 IP 주소는 각각의 MQA-가능 네트워크 디바이스를 식별하기 위해 애플리케이션에 의해 사용되고, 고유 포트 번호는 QoS-인식 애플리케이션 유일하게 정의하기 위하여 MQA에 의해 사용된다. 이처럼, 특정의 QoS-인식 애플리케이션과 하나 이상의 대응 네트워크 인터페이스 디바이스의 NQA 사이의 각 소켓은 고유하므로, MQA가 어느 다른 애플리케이션으로부터 요청 애플리케이션을 식별할 수 있고 구분할 수 있고, 그 요청 애플리케이션은 QAoS 메시지가 수신된 특정 MQA와 대응 네트워크 인터페이스 디바이스(들)를 식별할 수 있고 구분할 수 있다. 게다가, OS 경로 테이블은 애플리케이션 패킷이 항상 디폴트 경로 테이블 엔트리에 관계없이 또는 VPN이 사용되고 있는지에 관계없이 항상 적합한 MQA-가능 네트워크 디바이스로 경로설정되는 것을 보증한다.

<41> 일 예의 실시예에서, QoS API 메시징 프로토콜이 제공된다. 도 2는 QAoS 메시지 포맷의 일 예의 실시예를 도시한다. 메시지(400)는 인터넷 프로토콜 헤더(410), 전송 프로토콜 헤더(420), QAoS 헤더(430) 및 QAoS 데이터(440)를 포함할 수 있다. QAoS 헤더(430)는 QAoS 버전 번호(434)와 QAoS 프로토콜 부호(436)를 포함할 수 있으며, 이는 시스템에 의해 사용되는 QAoS 프로토콜의 특정 유형을 식별한다. 표시된 바와 같이, 프로토콜 메시지는 QAoS 패킷 헤더(430)에 캡슐로 넣어질 수 있다. 이 캡슐화 메커니즘은 애플리케이션과 MQA 사이에서 QAoS 메시징의 실제 데이터 페이로드 포맷의 사양과 협상(negotiation)을 허락한다. 이는 표준화되고 독점적인 QoS API 메시징 프로토콜들과 그런 프로토콜의 다중 개정판(multiple revisions)의 모두가 애플리케이션과 MQA 사이에서 QoS API 메시징의 전송을 위해 지원되는 것을 허용한다.

<42> 소켓이 QAoS에 관해 설정되고 QoS API 메시징 프로토콜 및 개정판이 MQA와 협상되면, 애플리케이션은 QoS 통지를 수신하고 QoS 요청을 송신하기 위해 QAoS를 사용할 수 있다. QoS 통지는 네트워크내 QoS의 다른 레벨과 유형의 현재 가용성에 대한 정보, 가용 및/또는 활동적인 네트워크 서비스 흐름의 설정, 서비스 흐름의 각각의 대응되는 가용 및/또는 능동 분류자(들), 네트워크에 의해 지원된 현재와 가용 서비스 흐름 데이터 표시 계획, 데이터와 다른 서비스의 어떤 정의 및/또는 능동 네트워크 필터링 또는 조절(throttling)을 포함할 수 있다. QoS 요청은 QoS 서비스 흐름, 분류자, 데이터 필터링, 파형 정형, 및 네트워크 인터페이스 디바이스(들)와 그 대응하는 네트워크(들)의 MAC 층의 데이터 조정 특징을 협상하고, 추가하고, 수정하고, 삭제하고, 부착하고, 탈착하고, 활성화하고, 비활성화하는데 사용될 수 있다.

<43> 또한 애플리케이션과 MQA 사이의 QAoS 링크는 QoS 속성 및 상응하고 있는 네트워크(들)의 상태 변경을 보고하기 위해 사용될 수 있다. 일 예의 실시예에서, 네트워크 디바이스는 네트워크 특성을 끊임없이 감시할 수 있고, 만일 이 특성이 변하여 동의한 QoS 서비스 레벨이 더 이상 성취될 수 없다면, QoS-인식 애플리케이션과 통신하고 QoS-인식 애플리케이션이 어떤 필요한 조치를 취하는지를 판단할 수 있도록 QoS-인식 애플리케이션에게 그 변화를 알려준다. QoS-인식 애플리케이션으로부터의 QoS 요청에 응답하여, 네트워크 디바이스는 네트워크 계층 태그 또는 네트워크 계층 메시지를 삽입/부착할 수 있으므로, 주어진 QoS-인식 애플리케이션에 의해 요청받은 네트워크 계층 QoS 메커니즘을 실현하기 위하여 하류의 네트워크 경로 디바이스에 지시한다.

<44> 일 예의 실시예에서, 네트워크 인터페이스 디바이스의 MQA는 하나의 쉽게 인식되고 고유한 멀티캐스트 소켓을 넘어, 방송 QoS API 이벤트 통지를 모든 듣고 있는 QoS-인식 애플리케이션으로 송신할 수 있다. QoS-인식 애플리케이션은 그런 멀티캐스트 QAoS 소켓을 등록할 수 있고 등록해제(de-register)할 수 있다. 다른 예 실시예에서, 애플리케이션은 오래된(stale) QAoS 소켓과 관련된 소켓의 획득한 네트워크 QoS 자원을 자유롭게 허용할 수 있을 때를 결정하기 위하여 MQA의 방법을 제공하기 위해 재등록할 수 있고 및/또는 MQA는 그 애플리케이션을 신호를 보내어 테스트(핑; ping)할 수 있다. 각종 대체 실시예에서, MQA는 당해 기술분야에서 숙련된 자들에게 알려져 있는 다른 함수를 제공할 수 있다.

<45> 여기에 기술된 시스템과 모듈은 소프트웨어, 펌웨어, 하드웨어 또는 여기에 기술된 목적에 적합한 소프트웨어, 펌웨어 또는 하드웨어의 어떤 조합(들)을 포함할 수 있다. 소프트웨어와 다른 모듈들은 서버, 워크스테이션, 개인용 컴퓨터, 전산화된 태블릿, PDA와 여기에 기술된 목적에 적합한 다른 디바이스에 존재할 수 있다. 소프트웨어와 다른 모듈은 국부 메모리를 경유하여, 네트워크를 경유하여, ASP 컨텍스트내의 브라우저 또는 다른 애플리케이션을 경유하여, 또는 여기에 기술된 목적에 적합한 다른 수단을 경유하여 액세스 가능하다. 여기에 기술된 데이터 구조는 컴퓨터 파일, 변수, 프로그래밍 배열, 프로그래밍 구조 또는 어떤 전자식 정보 기억 기획 또는 방법. 또는 그것과 목적에 적합한 어떤 조합을 포함할 수 있다.

<46> 실시예와 애플리케이션이 도시되고 기술되었지만, 위에 언급된 것보다 더 많은 변경예가 여기에 개시된 발명 개념에서 벗어남이 없이 가능하다는 것은 당해 기술에 숙련된 자들에게 명백할 것이다. 본 발명은 그러므로 첨부된 청구범위의 정신에 제한되는 것 이외에는 제한되지 않는다.

도면의 간단한 설명

<14> 본 발명은 발명의 실시예를 도시하기 위해 사용되는 다음 설명과 첨부도면을 참조하면 가장 잘 이해될 수 있다.

<15> 도면들 중에서:

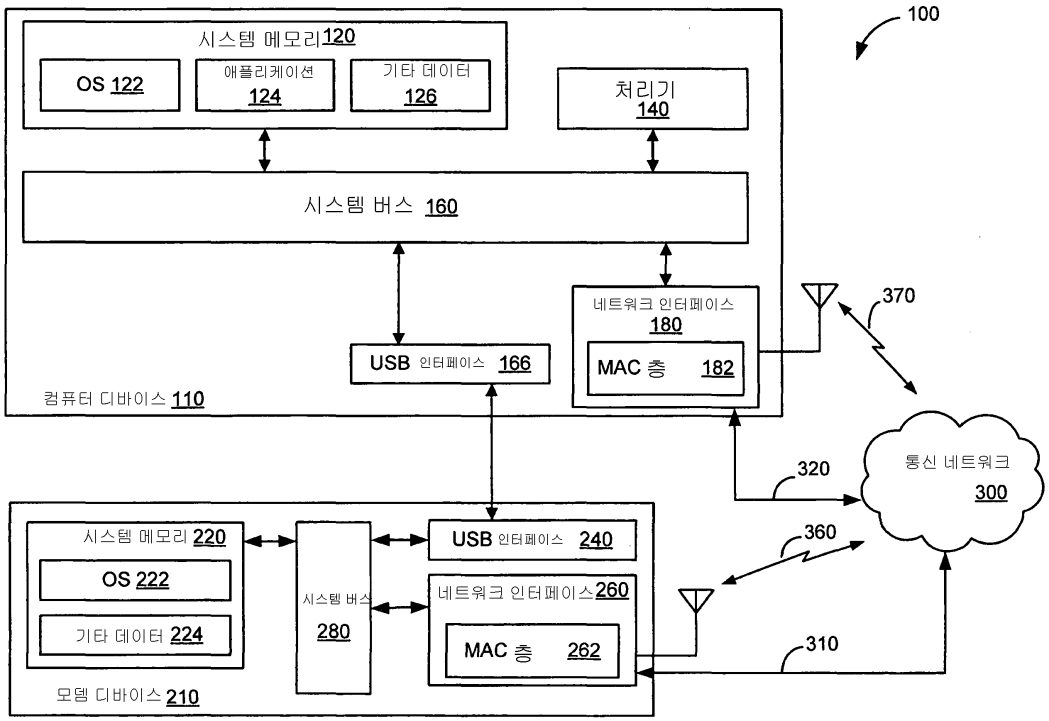
<16> 도 1은 네트워크 구조의 일 예의 실시예의 다이어그램이고;

<17> 도 2는 QoS 메시지의 일 예의 실시예의 다이어그램이고; 및

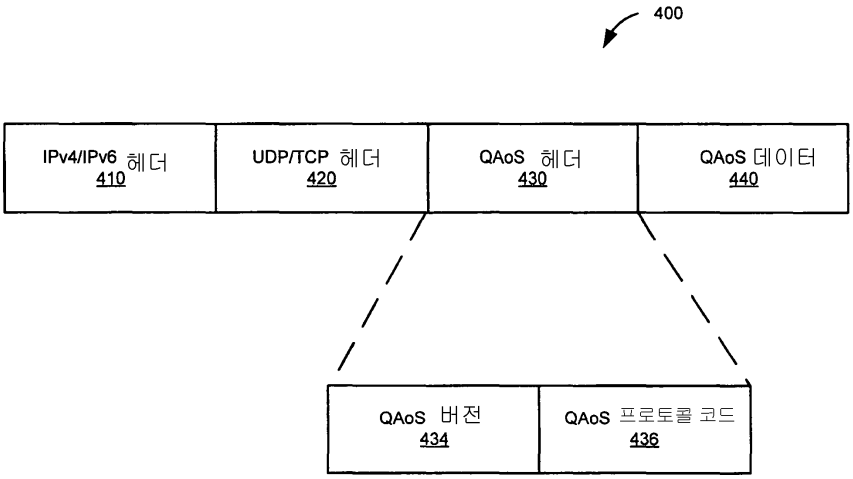
<18> 도 3은 QoS 제어 메커니즘의 일 예의 실시예의 다이어그램이다.

도면

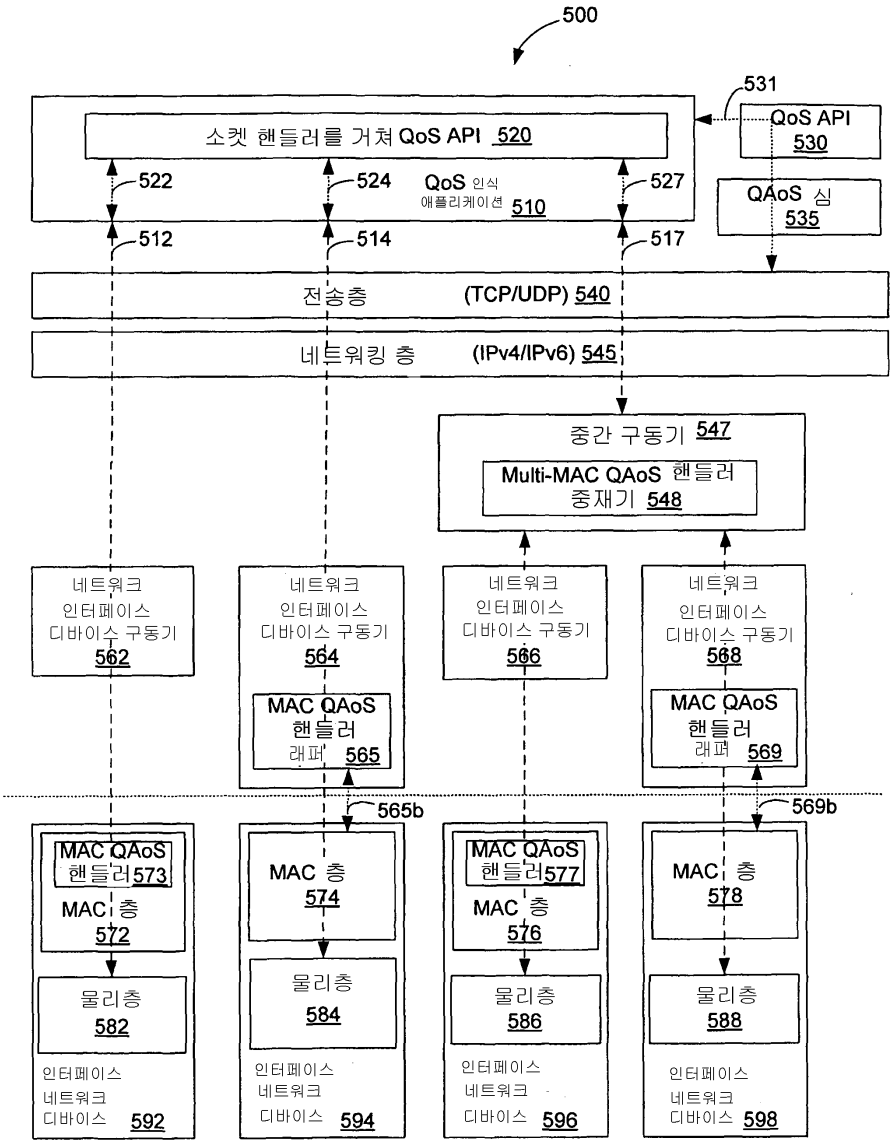
도면1



도면2


400
IPv4/IPv6 헤더
410
UDP/TCP 헤더
420
QAoS 헤더
430
QAoS 데이터
440
QAoS 버전
434
QAoS 프로토콜 코드
436

도면3


500
531
소켓 핸들러를 거쳐 QoS API 520
QoS API 530
522
524
QoS 인식 애플리케이션 510
527
QoS 심 535
512
514
517
전송층 (TCP/UDP) 540
네트워킹 층 (IPv4/IPv6) 545
중간 구동기 547
Multi-MAC QoS 핸들러 중재기 548
네트워크 인터페이스 디바이스 구동기 562
네트워크 인터페이스 디바이스 구동기 564
MAC QoS 핸들러 래퍼 565
네트워크 인터페이스 디바이스 구동기 566
네트워크 인터페이스 디바이스 구동기 568
MAC QoS 핸들러 래퍼 569
565b
569b
MAC QoS 핸들러 573
MAC 층 572
물리층 582
인터페이스 네트워크 디바이스 592
MAC 층 574
물리층 584
인터페이스 네트워크 디바이스 594
MAC QoS 핸들러 577
MAC 층 576
물리층 586
인터페이스 네트워크 디바이스 596
MAC 층 578
물리층 588
인터페이스 네트워크 디바이스 598