
A.1. AQUILA – Quality of Service für IP-Netzwerke¹

*Falk Fünfstück
Heinrich Hußmann
Anne Thomas*

1 Einleitung

Im Zuge sich immer stärker verbreitender Anwendungen und Dienste, die auf Internet-Technologien beruhen, wird eine grundlegende Einschränkung deutlich: Einerseits werden immer vielseitigere und leistungsfähigere Internet-Anwendungen geschaffen, andererseits bauen diese aber auf einer seit langem eingesetzten und bewährten Technologie auf – die des Internet-Protokolls und der darauf aufsetzenden Protokolle, deren Grundlagen vor mehreren Jahrzehnten gelegt wurden. Diese Protokolle entsprechen jedoch nicht mehr allen Anforderungen moderner Kommunikationsanwendungen.

Diese teilweise sehr hohen Anforderungen betreffen vor allem die Dienstgüte, die eine Netzwerkinfrastruktur – in unserem Falle das Internet – einer Anwendung individuell zusichern soll. Diese Dienstgüte wird als Quality of Service (QoS) bezeichnet und bezieht sich auf den Durchsatz des Netzwerkes, vor allem aber auf die Antwortzeit, die Schwankungen innerhalb der Antwortzeit und die Fehler- bzw. Verlustrate. Doch im Gegensatz zur neueren Technologien wie ATM² wurde das Internet nicht dazu konzipiert, solche QoS-Merkmale einer Verbindung anzubieten und zuzusichern. Jedweder Verkehr im Internet wird nach dem Prinzip „so gut es geht“ (engl. „best effort“) behandelt, was zu Einschränkungen bei der Benutzbarkeit einiger Anwendungen und Online-Dienste führen kann und auch führt. Ein Beispiel für eine fehlende Ende-zu-Ende-Dienstgüte (d.h. über alle Instanzen des Kommunikationsweges hinweg) ist der Börsengang der Infineon Technologies AG, der große Probleme beim Zugang zu Online-Banking-Diensten verursachte.

Das von der EU-geförderte IST³-Projekt AQUILA (Projektlaufzeit 2000-2002) hat das Ziel, eine Architektur zu entwickeln und prototypisch zu erproben, die das Internet um QoS-Merkmale erweitert. Existierenden und zukünftigen Internet-Anwendungen wird

¹ Diese Arbeit wurde gefördert von der Europäischen Union unter Projekt-Nr. IST-1999-10077: „AQUILA – Addaptive Resource Control for QoS Using an IP-based Layered Architecture”.

² ATM – Asynchronous Transfer Mode

³ IST – Information Society Technologies Programme

der Zugang zu neuen QoS-Diensten mittels einer im Rahmen des Projektes zu realisierenden Middleware ermöglicht [AQUILA].

Im Folgenden werden zunächst das technologische Umfeld des Projektes, insbesondere wichtige Anwendungsszenarien im Internet und die für AQUILA grundlegenden QoS-Technologien skizziert (Kapitel 2). Danach werden die Herangehensweise und die Gesamtarchitektur von AQUILA vorgestellt (Kapitel 3), bevor auf die konkrete Anwendungsunterstützung durch die Middleware, das End-user Application Toolkit (EAT), eingegangen wird (Kapitel 4). Das Dokument schließt mit einer Zusammenfassung und einem Ausblick (Kapitel 5).

2 Technologisches Umfeld

2.1 Anwendungsszenarien

Heutige Internet-Anwendungen basieren zunehmend auf multimedialen, umfangreichen Inhalten sowie auf interaktiven, zeit- und fehlerkritischen Kommunikationsbeziehungen. Dadurch steigen die Anforderungen an die zugrunde liegenden Infrastrukturen, indem sowohl der Umfang des zu übertragenden Datenverkehrs als auch der Bedarf nach kurzen Antwortzeiten und geringen Fehler- und Verlustraten wächst.

Eine grobe, nicht notwendigerweise vollständige Einteilung zeitgemäßer Internet-Anwendungen in Anwendungsgruppen mit ihren speziellen QoS-Anforderungen lässt sich wie folgt vornehmen:

Streaming Multimedia: Hier werden aufbereitete multimediale Daten, z.B. Video- und Audiodaten, von einem Server „heruntergeladen“ und im Client gepuffert, ehe sie dem Benutzer kontinuierlich präsentiert werden (Video-on-Demand, Audio-on-Demand). Durch die lokale Pufferung und abgleichende Präsentationsmechanismen ist diese Anwendungsgruppe relativ unempfindlich gegenüber Verzögerungen der Antwortzeit (Delay) und Varianzen dieser Verzögerungen (Jitter) im Netzwerk. Zudem zeichnet sich Video und mit Einschränkungen auch Audio durch eine geringere Verlust- und Fehlerempfindlichkeit aus. Das wichtigste Kriterium ist hier also der (mittlere) Durchsatz des Netzwerkes, um das Herunterladen in einer vertretbaren Zeit zu gewährleisten. Solche Szenarien basieren z.B. auf den Programmen RealPlayer, Apple QuickTime und Nullsoft WinAmp.

Interactive Multimedia: Im Gegensatz zu obiger Anwendungsgruppe kommt hier der Aspekt der direkten Interaktivität zwischen den Anwendungen ins Spiel. Das heißt, die zu übertragenden multimedialen Daten liegen nicht in aufbereiteter Form vor, sondern werden aufgrund einer Interaktion erst erstellt. Dementsprechend können sie im Client nicht mehr gepuffert werden, da sie sofort benötigt und präsentiert werden sollen.

Beispielanwendungen hierfür sind Multimediakonferenzen (z.B. via Microsoft NetMeeting) und Internet-Telefonie. Die Qualität ist abhängig vom Durchsatz, wobei das Datenaufkommen sehr unregelmäßig sein kann. Wichtige Rollen spielen auch Delay (Warten auf die Antwort) und vor allem Jitter (Varianzen der Wartezeiten).

Mission Critical: In diese Gruppe ordnen sich Anwendungen ein, die eine zuverlässige Übertragung ihrer (diskreten) Nachrichten benötigen, um erwartungskonform arbeiten zu können. Sie stellen besonders hohe Anforderungen an Delay und Fehler- bzw. Verlustrate des Netzwerkes. Es lassen sich hierfür zahlreiche typische Vertreter finden: Online-Banking bzw. -Brokerage, verteilte Datenbank- und Geschäftsanwendungen, etc. Der Durchsatz und die damit die benötigte Netzwerkbandbreite spielt wegen der i.A. geringen Nachrichtengröße eine untergeordnete Rolle.

Es lassen sich zahlreiche Mischanwendungen finden, die Merkmale mehrerer Gruppen aufweisen. So haben z.B. Online-Spiele sowohl einen interaktiven Multimedia- als auch einen missionskritischen Charakter. Die speziellen Anforderungen richten sich nach den verwendeten Medientypen und Kommunikationsmodellen.

Die oben genannten Qualitätsanforderungen an eine Kommunikationsverbindung sind jedoch anwendungsbezogen dargestellt. In diesem Zusammenhang ist zu beachten, dass sich die einzelnen QoS-Merkmale einer Verbindung auch gegenseitig bedingen: So führt beispielsweise eine hohe Verlustrate bei der Verwendung von TCP⁴ zu einem wiederholtem Senden der Daten und damit zum Anstieg von Delay bzw. zum Absinken des (effektiven) Durchsatzes.

2.2 Existierende QoS-Technologien für das Internet

Einige Ansätze versuchen derzeit, dem existierenden Internet QoS-Fähigkeiten zu verleihen. Die zwei bedeutendsten sind Integrated Services/RSVP (IntServ) und Differentiated Services (DiffServ) der IETF⁵:

Integrated Services: Das IntServ-Modell basiert auf einer Erweiterung des bestehenden Internets um neue Dienste: Guaranteed mit harten Garantien bzgl. Bandbreite und Delay für echtzeitintolerante Anwendungen (z.B. Interactive Multimedia) sowie Controlled Load mit weniger harten Garantien bzgl. Delay für echtzeitintolerante Anwendungen (z.B. Streaming Multimedia). Wesentlicher Bestandteil des Modells ist das empfangenorientierte Resource Reservation Protocol (RSVP), das die Ressourcen zwischen Sender und Empfänger im Voraus reserviert. Auf Grund ungelöster Probleme vor allem im Weitverkehrsbereich (hoher Protokoll-Verwaltungsaufwand, ungelöstes Skalierbarkeits-

⁴ TCP – Transmission Control Protocol

⁵ IETF – Internet Engineering Task Force

problem) wird der Einsatz von RSVP zunehmend für lokale Netze zur expliziten Signalisierung von QoS-Anforderungen diskutiert [Black99].

Differentiated Services: Im Gegensatz zu IntServ kommt das DiffServ-Modell ohne zusätzliches Signalisierungsprotokoll aus, indem ein bislang fast ungenutztes Byte (Type-of-Service) des IP-Headers genutzt wird, um IP-Pakete hinsichtlich ihrer Verkehrsklassen zu klassifizieren und damit eine Priorisierung innerhalb der Router zu ermöglichen. Hierfür wurden zwei „Behandlungsregeln“ für Datenverkehr (Per-Hop Behavior) definiert: Expedited Forwarding mit zugesicherter Datenrate und einer Verkehrsklasse höchster Priorität sowie Assured Forwarding mit vier Verkehrsklassen mit bestimmten Verlustwahrscheinlichkeiten. Ein wichtiger Nachteil von DiffServ ist, dass im Gegensatz zu IntServ/RSVP keine Ende-zu-Ende-Garantien vergeben werden können. DiffServ ist vor allen für den Weitverkehrsbereich geeignet; häufig wird deswegen auch die Kombination RSVP (LAN) und DiffServ (WAN) diskutiert [Paul99].

Neben diesen beiden Modellen existieren noch eine Reihe weiterer, von den an dieser Stelle nur das *Multi-Protocol Label Switching* (MPLS) Erwähnung finden soll. Informationen zu diesen und weiteren gängigen QoS-Technologien können in [Stardust] gefunden werden.

Darüber hinaus wird mit dem *Internet2* eine teilweise neue Technologie entwickelt, die von Anfang an QoS-Aspekte mit berücksichtigt. Das Internet 2 basiert auf DiffServ als der zur Zeit erfolgsversprechendsten Idee. Um Garantien für QoS-Reservierungen vergeben zu können, wird das Konzept des Bandwidth Brokers eingesetzt [QBBAC].

Zusammenfassend ist zu sagen, dass alle hier vorgestellten Ansätze auf Protokollebene arbeiten. Um aber den eigentlichen Internet-Anwendungen die QoS-Fähigkeiten des Netzwerkes zugänglich zu machen, müssen die Anwendungen entweder selbst diese Protokolle beherrschen, oder es wird eine zusätzliche, vermittelnde Schicht, z.B. eine Middleware, zwischen Anwendungsebene und Protokollebene eingefügt. (Weiterführende Arbeiten zu diesem Thema sind in [Aurrec98] zu finden.)

3 Das AQUILA-Projekt

3.1 Hintergrund

Das AQUILA-Projekt benutzt und erweitert die bestehenden QoS-Technologien zur Entwicklung einer skalierbaren, effizienten und kompatiblen QoS-Architektur für das existierende Internet. Essentieller Part ist die benutzerfreundliche Bereitstellung von *Netzwerkdiensten*, die QoS-Eigenschaften aufweisen. Der Endbenutzer soll über die Möglichkeit verfügen, sich die zu einem Zeitpunkt angebotenen Dienste des Netzwerkes anzusehen und hierfür Verträge abzuschließen. Anschließend soll er dazu in der

Lage sein, für seine Anwendungen diese Netzwerkdienste in Anspruch zu nehmen. Das AQUILA-Netzwerk sichert den Endbenutzern eine Ende-zu-Ende-Dienstgüte zu. Dabei werden deren speziellen Bedürfnisse und Anwendungen bei der Konfiguration und Bereitstellung der entsprechenden Netzwerkdienste berücksichtigt. Durch Messungen der erreichten Dienstqualität werden Rückschlüsse auf die Zuteilung von Netzwerkressourcen ermöglicht. Die Architektur wird ferner in mehreren Feldversuchen, die auf realen Internet-Szenarien beruhen, getestet und evaluiert. Das Projekt entwickelt für diese Versuche Prototypen für die angestrebte QoS-Architektur.

3.2 Architektur

Die Architektur basiert auf einem 3-Schichtenmodell, das in Abbildung 1 dargestellt ist. AQUILA konzentriert sich auf die Entwicklung einer neuen Schicht, der *Resource Control Layer (RCL)*, die die globalen Ressourcen der unterliegenden IP⁶-Schicht – die die DiffServ-Technologie einsetzt – überwacht und verteilt. Sie übernimmt und erweitert die Funktionen eines verteilten Bandwidth Brokers. Die RCL besteht im Wesentlichen aus einer (erweiterbaren) Anzahl verschiedener Agenten mit klar abgegrenzten Verantwortungsbereichen. Diese Schicht ist jedoch unabhängig von der physikalischen Infrastruktur.

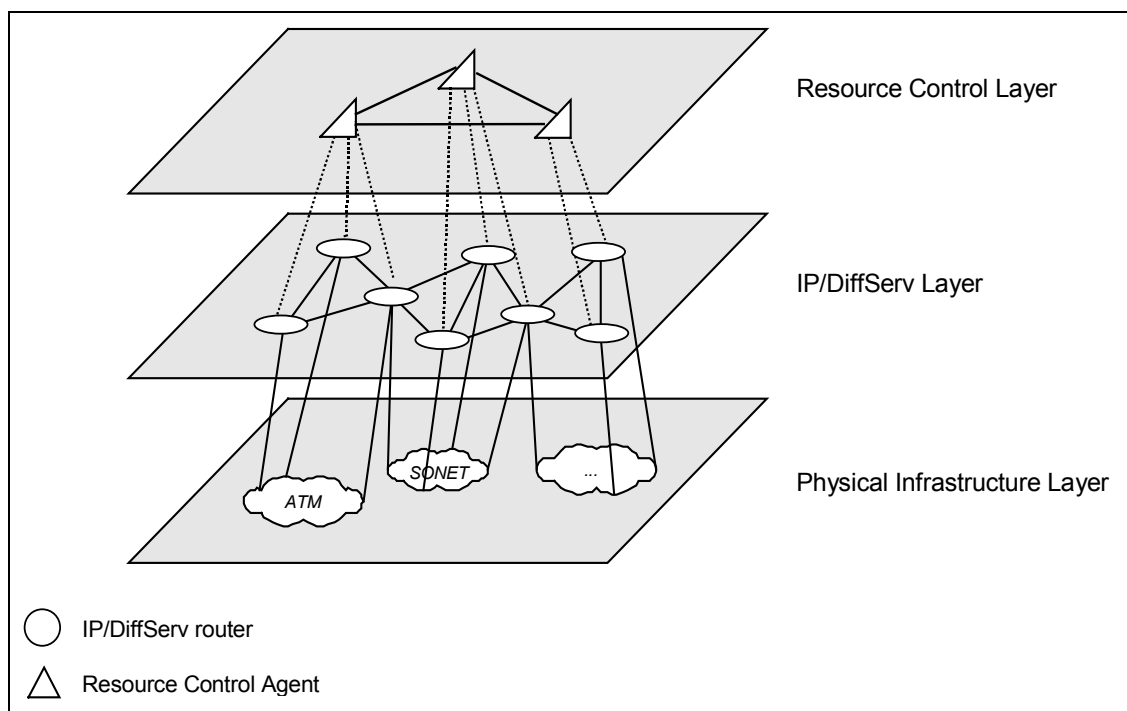


Abbildung 1: 3-Schichtenmodell von AQUILA [Koch00]

⁶ IP – Internet Protocol

Das physikalische AQUILA-Netzwerk ist in mehrere Komponenten unterteilt (Abbildung 2). Die Endbenutzer-Terminals (*Hosts*) sind via Zugangsnetzwerke (*Access Networks*, z.B. LAN) an ein Kernnetzwerk (*Core Network*, z.B. WAN) angeschlossen. Dies geschieht mittels so genannter *Edge Router (ER)*. Innerhalb des Kernnetzwerkes übernehmen *Core Router (CR)* das Routing. Das Kernnetzwerk kann in mehrere ISP⁷-Domänen aufgeteilt sein. In diesem Falle übernehmen *Border Router (BR)* das Routing zwischen zwei Domänen. Alle genannten Arten von Routern beherrschen die DiffServ-Technologie. Diese Architektur ist die Basis für die Skalierbarkeit des AQUILA-Konzeptes [Ferguson98].

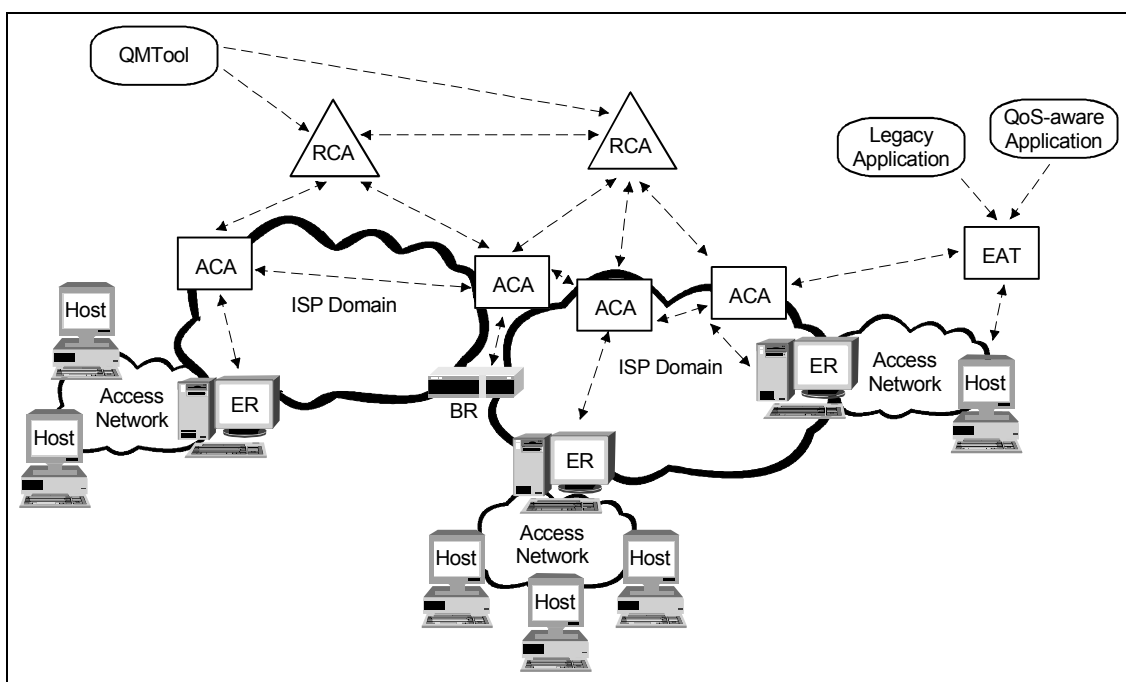


Abbildung 2: Netzwerkarchitektur von AQUILA [Winter00]

Logisch setzt auf diese Infrastruktur die bereits erwähnte RCL auf, die ihrerseits aus mehreren Komponenten besteht:

Resource Control Agents (RCA): Ein RCA überwacht und verteilt, über mehrere Hierarchieebenen hinweg, die Ressourcen (Bandbreiten) einer Domäne. Er agiert damit als Bandwidth Broker. Ein RCA kann mit einem anderen RCA kommunizieren, um QoS-Unterstützung über Domänengrenzen hinweg zu ermöglichen. Darüber hinaus kommuniziert ein RCA mit den Admission Control Agents seiner Domäne, um diesen Ressourcen zuzuteilen.

⁷ ISP – Internet Service Provider

Admission Control Agents (ACA): Ein ACA verwaltet jeweils einen Edge oder Border Router. Dafür bekommt er vom RCA die lokalen Ressourcen des Routers zugeordnet. Ein ACA kann mit einem anderen ACA derselben Domäne kommunizieren, um eine QoS-Reservierung innerhalb der Domänengrenzen zu ermöglichen. Ferner kommuniziert ein ACA mit den zugeordneten End-user Application Toolkits, um diesen Netzwerkdienste anzubieten und hierfür Reservierungen entgegenzunehmen. Für eine Reservierung werden eine lokale Admission Control (Sind genügend Ressourcen vorhanden?) und Policy Control (Hat der Nutzer die Erlaubnis dafür?) durchgeführt.

End-user Application Toolkits (EAT): Ein EAT bildet die Middleware zwischen den Anwendungen der Endbenutzer und dem AQUILA-Netzwerk bzw. der RCL. Im Auftrag der Endbenutzer werden Reservierungen für Netzwerkdienste bei dem verantwortlichen ACA angefordert. Diese Reservierungen beziehen sich auf vom EAT verwaltete Hosts (Sender oder Empfänger), wobei sowohl sender- als auch empfängerorientierte Szenarien ermöglicht werden. Ebenso werden Punkt-zu-Punkt- oder Punkt-zu-Mehrpunkt-Reservierungen gestattet. Das EAT unterstützt Legacy- und QoS-aware Applications (Anwendungen) (siehe nächstes Kapitel).

Die Komponente QMTool⁸ aus Abbildung 2 wird hier nur kurz angesprochen: Sie ist ein Werkzeug, das dem Netzwerkoperator die Konfiguration, Verwandlung und Überwachung des Netzwerkes sowie dessen Dienste gestattet.

Keine Aussage wird darüber getroffen, wo die logischen RCL-Komponenten (RCA, ACA, EAT) physikalisch angesiedelt werden. Es besteht lediglich die logische Beziehung $RCA \Leftrightarrow \text{Domäne}$, $ACA \Leftrightarrow \text{Edge/Border Router}$ und $EAT \Leftrightarrow \text{Host}$, die natürlich eine physikalische Verteilung nahelegt.

Zur Kommunikation zwischen den einzelnen Komponenten der RCL kommt ausschließlich CORBA⁹ zur Einsatz. Dies erlaubt eine maximale Flexibilität und einfache Erweiterbarkeit für den Nachrichtenaustausch, der damit nicht den Beschränkungen der aus der Netzwelt bekannten Reservierungs- (z.B. RSVP) und anderer Protokolle unterliegt. Darüber hinaus werden – um maximale Plattformneutralität hinsichtlich der für die RCL verwendeten Laufzeitumgebungen zu erreichen – alle Komponenten der RCL in Java implementiert. Analyse und Entwurf der Komponenten basieren konsequent auf UML¹⁰.

Im Folgenden wird auf das EAT, auf die zur Anwendungsunterstützung wichtigsten Komponente, näher eingegangen.

⁸ QMTool – QoS Management Tool

⁹ CORBA – Common Object Request Broker Architecture

¹⁰ UML – Unified Modelling Language

4 EAT als Middleware zur Anwendungsunterstützung

4.1 Anforderungen und Funktionalität

Das End-user Application Toolkit dient als Middleware zwischen der Netzwerk-Infrastruktur und den Endbenutzer-Anwendungen und hat demzufolge Anforderungen in beiden Richtungen zu erfüllen: Für die Resource Control Layer des AQUILA-Netzwerkes agiert das EAT als Front-end Tool; für die Anwendungen als (alleiniges) QoS-Portal zum AQUILA-Netzwerk.

Gegenüber der RCL, insbesondere dem ACA, besitzt das EAT eine wohldefinierte Schnittstelle, deren Grundlage eine begrenzte Anzahl von Netzwerkdiensten ist. Für diese Dienste werden Ressourcenanforderungen in Form von Reservierungen getätigt. Für das EAT ist dies die einzige Kommunikationsmöglichkeit mit der RCL; insbesondere hat das EAT keine Schnittstelle zum RCA.

Gegenüber den Anwendungen sind die Anforderungen wesentlich breiter gestreut. Es ist ein Ziel des Projektes, möglichst viele unterschiedliche Anwendungen inklusive ihrer unterschiedlichen Eigenschaften zu unterstützen. Drei Klassen von zu unterstützenden Anwendungen können ausgemacht werden:

Legacy Applications: Zu dieser Klasse gehören alle existierenden Internet-Anwendungen, die bisher noch keine QoS-Funktionalität aufweisen bzw. keine QoS-Technologien unterstützen. Es wird angestrebt, zumindest einige, weit verbreitete Applikationen zu unterstützen, z.B. herkömmliche WWW-Browser, das Konferenzsystem NetMeeting, Streaming-Anwendungen wie der RealPlayer, IP-Telefonie-Software und interaktive Online-Spiele für mehrere Benutzer.

Hierfür stellt das EAT in der ersten Projektphase Benutzungsschnittstellen bereit, in denen die QoS-Reservierungen (manuell) spezifiziert werden können. Für bestimmte Anwendungen ist des weiteren die Verwendung von Proxies notwendig, um deren Steuerungsdaten auszuwerten.

QoS-aware Applications: In diese Klasse ordnen sich existierende oder zukünftige Anwendungen ein, die QoS-Technologien wie z.B. RSVP beherrschen. Interessante Kandidaten sind beispielsweise neue Applikationen (neue Version von NetMeeting, etc.), die das QoS-API von Windows 2000 benutzen.

Für diese Anwendungen werden z.B. Proxy- und Dämon-Mechanismen bereitgestellt.

EAT-based Applications: Darunter sind all jene Anwendungen zu verstehen, die die QoS-Fähigkeiten des AQUILA-Netzwerkes mittels des EAT direkt und transparent nutzen können und sich damit ebenfalls als „QoS-aware“ klassifizieren lassen. Dies können neue, aber auch Modifikationen bereits existierender Anwendungen sein, z.B.

basierend auf dem Java Media Framework. Sie bilden hinsichtlich der Anwendungsunterstützung das Hauptaugenmerk des AQUILA-Projektes.

Das EAT stellt ein API¹¹ für diese Anwendungen bereit, so dass sie ihre QoS-Anforderungen automatisch anzeigen können. Diese Funktionalität wird jedoch erst zu einem späteren Zeitpunkt realisiert.

Für alle Anwendungen gilt, dass den Endbenutzern Individualisierungsmöglichkeiten angeboten werden, diese Internet-Anwendungen an ihre konkreten Bedürfnisse anzupassen. (Unter Endbenutzern sollen in diesem Zusammenhang alle Benutzer des AQUILA-Netzwerkes verstanden werden, sowohl die Anbieter als auch die Konsumenten von Online-Anwendungen und -Inhalten.) So sollen sie beispielsweise dazu in der Lage sein, die Bildgröße und die Bildqualität eines Videokonferenzsystems festzulegen und deren einwandfreie Übertragung anzufordern, oder sie sollen für eine Online-Banking-Sitzung eine sichere und schnelle Kommunikation anfordern können.

4.2 Architektur

Das EAT ist modular aufgebaut und besteht aus mehreren, unter Umständen verteilten Komponenten, die zudem von Plattform zu Plattform variieren können. Die Grundfunktionalität ist jedoch auf allen Plattformen gleich; lediglich einzelne Bausteine (z.B. bestimmte Proxies) können ausgetauscht werden.

Das EAT bietet mehrere Arten von wichtigen Schnittstellen an: die mittels IDL¹² spezifizierte CORBA-Schnittstelle zum ACA, die Proxies und ein API für die Anwendungen sowie Benutzungsschnittstellen für die Endbenutzer. Diese Schnittstellen sind steuerflussorientiert. Des Weiteren hat das EAT eine datenflussorientierte Schnittstelle zum Edge Router (ER). Ebenfalls vorhanden ist eine Datenbankverbindung zum Zugriff auf vorbereitete Profiles (siehe Abschnitt 4.4).

¹¹ API – Application Programming Interface

¹² IDL – Interface Definition Language

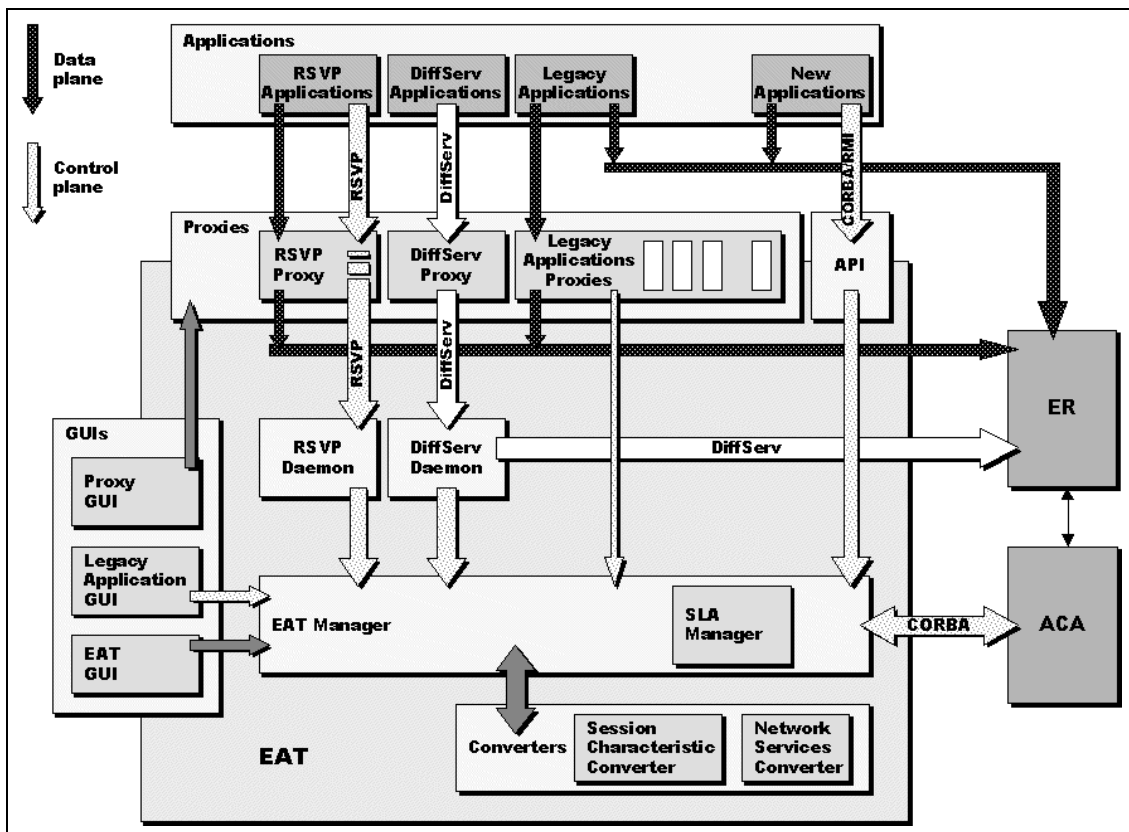


Abbildung 3: EAT-Architektur

Abbildung 3 zeigt den schematischen Aufbau des EAT. Wichtige interne Komponenten sind der *EAT Manager* und die *Converters*. Der *EAT Manager* ist die zentrale Komponente für Nutzerverwaltung und Reservierungsanforderung. Unter anderem verwaltet er die Verträge (SLA¹³) der Endbenutzer mit dem ISP. Speziell für die Reservierung benötigt er jedoch neben den Steuerinformationen der Anwendungen (z.B. IP-Adressen und Portnummern) weitere Informationen aus den zur Verfügung stehenden Netzwerkdiensten und den konkreten Anwendungsanforderungen an die Dienstgüte. Die Aufbereitung dieser Informationen ist die Aufgabe der Konverter, deren Funktionen im Folgenden erläutert wird.

4.3 Abbildung von QoS-Merkmalen

Die Schnittstelle zwischen EAT und ACA zur Reservierungsanforderung beruht auf Netzwerkdiensten (*Network Services, NS*). Diese sind gleichsam die Produkte des Netzwerkes und werden durch technische Parameter realisiert (z.B. detaillierten Aussagen darüber, welche Delay- und Jitter-Eigenschaften der konkrete Netzwerkdienst hat). Bei einer Reservierung für einen dieser Dienste müssen konkrete Angaben unter

¹³ SLA – Service Level Agreement

anderem über die benötigte Bandbreite übergeben werden; in ähnlicher Weise, wie dies bei RSVP geschieht. Die Netzwerkdienste sind wegen ihrer begrenzten Anzahl aber nur grob an konkrete Anforderungserfordernisse angelehnt. So entsprechen die bereitgestellten Netzwerkdienste in etwa der in Abschnitt 2.1 identifizierten Anwendungsgruppen Streaming Multimedia, Interactive Multimedia und Mission Critical. Zusätzlich existiert ein Standarddienst (best effort) für alle sonstigen Anwendungen (z.B. E-Mail). Diese grobe Unterteilung macht es erforderlich, dass die konkreten und teilweise sehr verschiedenen Anforderungserfordernisse auf die zur Verfügung stehenden Dienste abgebildet werden. Diese Funktion wird vom *Network Services Converter* übernommen. Seine Aufgabe liegt darin, für eine Anwendung, die bestimmte Anforderungen an Durchsatz, Delay, etc. stellt, einen oder mehrere passende Netzwerkdienste zu finden, für die der Endbenutzer einen gültigen Vertrag (SLA) abgeschlossen hat. Darüber hinaus werden die technischen Parameter (*Technical Parameters*) der Reservierungsanforderung für einen konkreten Netzwerkdienst spezifiziert.

Der Endbenutzer soll nun die Möglichkeit haben, aus der Liste von möglichen verschiedenen Qualitätsstufen (Dienstgütestufen), die mittels oben genannter Abbildungsfunktion realisiert werden können, auszuwählen. Hierfür müssen jedoch die spezifizierten technischen Parameter in für den Endbenutzer leicht verständliche Aussagen umgewandelt werden, da nicht davon ausgegangen werden kann, dass genügend technisches Verständnis vorliegt. Zum Beispiel können verschiedene Audioqualitäten angeboten werden: CD-, Radioqualität usw. (Hinzu kommt, dass der Benutzer auf die unterschiedlichen Kosten der verschiedenen Qualitätsstufen hingewiesen werden muss, da die Netzwerkdienste mit QoS-Merkmalen i.A. preisintensiver sind.) Diese zweite Abbildungsfunktion ist Aufgabe des *Session Characteristics Converter*. Die *Session Characteristics* bezeichnen dabei die dem Endbenutzer präsentierten Qualitätsstufen für eine Anwendungssitzung. Sie stehen für die (dem „normalen“ Endbenutzer nicht zugänglichen) technischen Parameter der Qualitätsstufen, die für die QoS-Reservierung benötigt werden.

Beide Abbildungsverfahren sind Bestandteil eines mehrstufigen Verfahrens zur Übersetzung von QoS-Parametern unterschiedlicher Abstraktionsniveaus (Abbildung 4; auf Application Profiles wird in nächsten Abschnitt näher eingegangen).

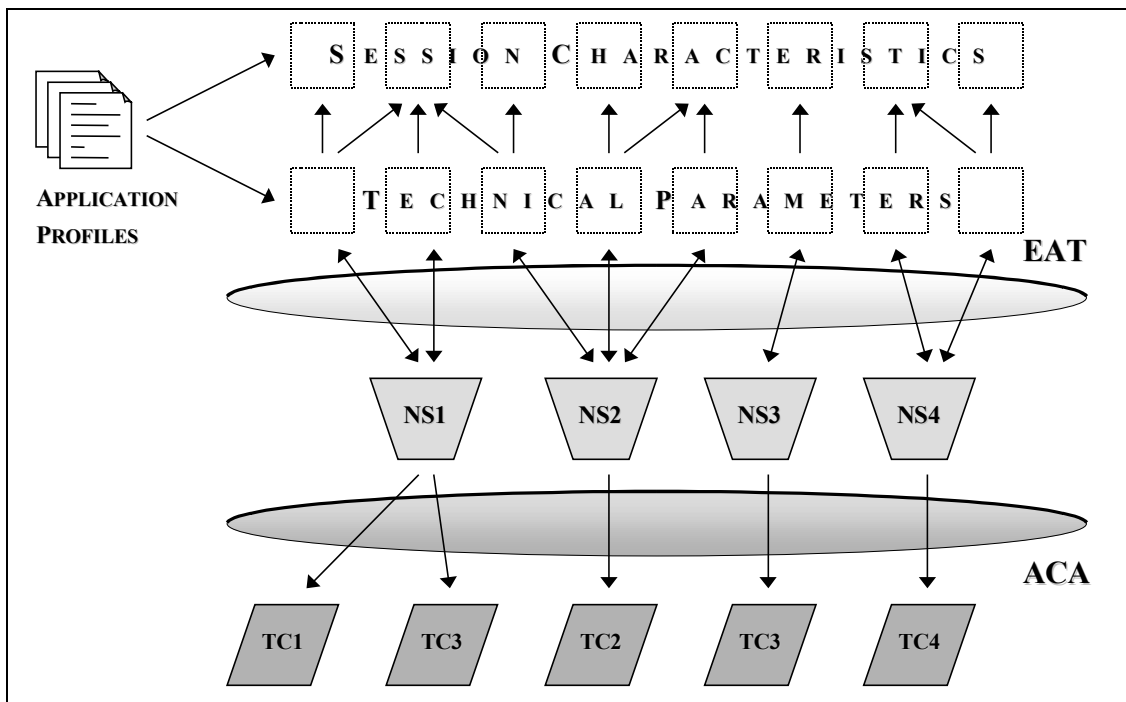


Abbildung 4: Mehrstufige Abbildung von QoS-Parametern

Das Abbildungsverfahren setzt sich innerhalb des ACA fort: Die angeforderten Netzwerkdienste (NS_x) werden letztlich durch konkrete Verkehrsklassen (*Traffic Classes*, TC_y), die ein bestimmtes Verhalten innerhalb des DiffServ-Netzwerkes repräsentieren, implementiert. Da jedoch das EAT völlig unabhängig von dieser Implementierung agiert und nur die Netzwerkdienste kennt, wird hier nicht näher darauf eingegangen. (Ebenso kennt der ACA nicht die konkreten Anwendungserfordernisse.)

Dieses verteilte, mehrstufige Abbildungsverfahren weist mehrere Vorteile auf: Einerseits ist es dem EAT freigestellt, die Abbildung der Anwendungserfordernisse auf unterschiedliche Arten zu realisieren. Damit kann flexibel auf neue Anwendungen reagiert werden, vorausgesetzt die Netzwerkdienste sind hinreichend flexibel und allgemeingültig definiert. Andererseits ist auch dem Netzwerkoperator freigestellt, die Netzwerkdienste auf individuelle Art und Weise, z.B. durch eine oder mehrere Verkehrsklassen zu implementieren, um beispielweise flexibel auf geänderte Netzwerkkonfigurationen reagieren zu können. So können in verschiedenen Domänen unter Umständen unterschiedliche Verkehrsklassen existieren, während sich die Netzwerkdienste gleichen.

4.4 Anwendungsprofile

Es wurde bereits festgestellt, dass Legacy-Anwendungen über keinerlei Möglichkeit verfügen, ihre QoS-Anforderungen selbst anzuzeigen. Das heißt, es muss ein Weg gefunden werden, diese Anforderungen separat zu spezifizieren und den Anwendungen

zuzuordnen. Darüber hinaus müssen in Abhängigkeit vom gewählten Anwendungstyp Vereinbarungen getroffen werden, wie die Session Characteristics letztlich dem Endbenutzer präsentiert werden, wie ihm beispielsweise die Auswahl einer Audioqualität ermöglicht werden soll.

Beides geschieht in AQUILA mittels so genannter Anwendungsprofile (*Application Profiles*). Ein Anwendungsprofil besteht aus einer Menge von Entwurfparametern für die Session Characteristics und einer Menge technischer Parameter. Es ist genau einer Applikation zugeordnet; beispielsweise existiert ein separates Profil für NetMeeting Version x.yz.

$$application_profile = \{session_characteristics\} \cup \{technical_parameters\}$$

Diese konkreten Parameter sind ihrerseits das Ergebnis generischer Parameter, die aus zwei Arten von generischen Profilen stammen: *Session Characteristics Profiles* und *Technical Profiles*:

$$session_characteristics_profile = \{session_characteristics\}$$

$$technical_profile = \{technical_parameters\}$$

Profile für die Session Characteristics enthalten die Entwurfparameter einer Benutzungsoberfläche für die Auswahl der Qualitätseigenschaften eines Anwendungstyps bzw. eines Medientyps, der zu übertragen ist. Beispielsweise stehen für eine Audioanwendung die Parameter CD-Qualität, Radioqualität usw. in dem entsprechenden Profile. Technische Profile enthalten technische Beschreibungen der von den Anwendungen benutzten Kompressionsverfahren, Codecs usw. der entsprechenden Medientypen. Beispielsweise werden hier die Bandbreitenanforderungen verschiedener Audioformate, wie z.B. MP3, spezifiziert. (Es ist vorgesehen, eine Art Profilbibliothek aufzubauen, z.B. mittels einer Datenbank, die ein breites Feld von Anwendungstypen und Verfahren unterstützt.)

Ein konkretes Anwendungsprofil ergibt sich dann aus der Zuordnung zwischen den relevanten Parametern aus den generischen Profilen und einer Anwendung, die definiert ist durch ihren Typ und die verwendeten Medien bzw. Verfahren zur deren Verarbeitung und Übertragung. Aufgrund der vielfältigen Datenstrukturen, die die einzelnen Parameter haben können, bietet sich XML¹⁴ als Beschreibungssprache für die Anwendungsprofile an. Damit wird auch die für neue Anwendungen und Technologien benötigte Flexibilität sichergestellt.

Die Verwaltung und Zuordnung der Profile ist Aufgabe des EAT bzw. der oben genannten Konverter. Für Legacy-Anwendungen muss das entsprechende Profil

¹⁴ XML – eXtensible Markup Language

manuell per Benutzungsschnittstelle ausgewählt werden, um dem EAT mitzuteilen, für welche Anwendung QoS angefordert werden soll. Für neue Anwendungen, die auf dem EAT basieren, wird eine automatische Auswahl angestrebt.

5 Zusammenfassung und Ausblick

Das vorliegende Dokument präsentierte den derzeitigen Status des AQUILA-Projektes, insbesondere die geplante Bereitstellung von Netzwerkdiensten mit definierter Dienstgüte (Quality of Service) für Internet-basierte Anwendungen. Es wurde auf das Problem hingewiesen, dass zwar geeignete QoS-Technologien für das Internet existieren, aber heutige Anwendungen nicht oder kaum von diesen Möglichkeiten Gebrauch machen können.

Das Projekt hat sich eine Architektur zum Ziel gesetzt, die existierende QoS-Technologien einsetzt und durch eine die Netzinfrastruktur überlagernde, logische Resource Control Layer eine garantierte Ende-zu-Ende-Dienstgüte ermöglicht. Darüber hinaus wird mittels des End-user Application Toolkits, das eine Middleware darstellt, sowohl existierenden, nicht-QoS-fähigen als auch neuen, QoS-fähigen Anwendungen ein Zugang zu den QoS-Features des AQUILA-Netzwerkes eröffnet.

Der vorgeschlagene Ansatz beruht unter anderem auf einem mehrstufigen Abbildungsverfahren von QoS-Parametern unterschiedlicher Abstraktionsniveaus. Damit wird eine maximale Flexibilität hinsichtlich neuer Anwendungen und Technologien sichergestellt. Der Einsatz von Anwendungsprofilen erlaubt darüber hinaus die Unterstützung möglichst vieler Anwendungsklassen und einen leichten Zugang für technisch nicht versierte Endbenutzer.

Die angestrebte Architektur wird in zwei Feldversuchen evaluiert. Der erste findet März 2001 statt, der zweite gegen Ende 2002 mit Ablauf der dreijährigen Projektdauer. Hierfür entwickelt das Projekt Prototypen der vorgestellten Komponenten, um die Praktikabilität der Herangehensweise zu untersuchen.

6 Literatur

- [AQUILA] AQUILA – Adaptive Resource Control for QoS Using an IP-based Layered Architecture, <http://www-st.inf.tu-dresden.de/aquila/>
- [Aurrec98] Cristina Aurrecochea, Andrew Campbell, Linda Hauw: A Survey of Quality of Service Architectures; ACM/Springer Verlag Multimedia Systems Journal, Special Issue on QoS Architecture, 3/1998, <http://www.ctr.columbia.edu/~campbell/andrew/publications/papers/survey.ps.gz>

-
- [Black99] Darryl P. Black: Building Switched Networks – Multilayer Switching, QoS, IP Multicast, Network Policy, and Service Level Agreements; Addison-Wesley, 1999
- [Ferguson98] Paul Ferguson, Geoff Huston: Quality of Service – Delivering QoS on the Internet and in Cooperate Networks; Wiley Computer Publishing, 1998
- [Koch00] Bert Koch et al.: Project Presentation; AQUILA Deliverable D002, 2000, <http://www-st.inf.tu-dresden.de/aquila/files/pub/d002-b0.pdf>
- [Paul99] Arindam Paul: QoS in Data Networks – Protocols and Standards; ftp://ftp.netlab.ohio-state.edu/pub/jain/courses/cis78899/qos_protocols/index.html
- [QBBAC] The Internet2 QBone Bandwidth Broker Advisory Council <http://www.internet2.edu/qos/qbone/QBBAC.shtml>
- [Siqueira] Frank Siqueira: A Framework for Distributed Multimedia Applications based on CORBA and Integrated Services Networks; Ph.D. Project, <http://www.cs.tcd.ie/Frank.Siqueira/PhD-Project/>
- [Stardust] QoS protocols & architectures at Stardust.com <http://www.stardust.com/qos/whitepapers/protocols.htm>
- [Winter00] Martin Winter et al.: System architecture and specification for first trial; AQUILA Deliverable D1201, 2000