CMnet - Kontinuierliche Medien im Netz verteilt

Manfred Lang, Robert Zwickenpflug

Technische Universität München, Lehrstuhl für Mensch-Maschine-Kommunikation,
Arcisstr. 21, D-80290 München,
zpg@mmk.e-technik.tu-muenchen.de

1 Intention

Heute neu auf den Markt erscheinende Workstations sind in der Regel für Multimedia-Applikationen vorbereitet. I/O-Möglichkeiten im Audiobereich wie Mikrofon und Lautsprecher sind zum Standard geworden. Ursprünglich zur leichteren Software-Distribution gedacht, haben CDROM-Laufwerke Zugang zur Computerwelt gefunden und auch Video-I/O-Systeme wie Kamera und entsprechende Rekorder werden heute als Zubehör angeboten, wenn sie nicht schon zur Grundausstattung gehören.

So gut es auf der Hardwareseite aussieht, so schlecht sieht es auf der Softwareseite aus. Zum Standard gehören oft nur einfache Routinen oder isolierte Einzelprogramme zur Benutzung der einzelnen Komponenten.

Im Entwicklungsstadium befinden sich derzeit zum einen lokale Singleuser-Entwicklungsumgebungen [Ang91] und zum anderen einfache Serverlösungen [Aro92, Dec93]. Eine standardisierte Integration von Audio in das X-Window-System ist erst für zukünftige Versionen geplant.

Wir implementieren derzeit eine multiuserfähige verteilte Entwicklungsumgebung für Continous-Media Applikationen, mit der es gelingen soll, Multimediadienste beliebig im Netz zu verteilen. Im Prinzip ist dies mit herkömmlichen Client-Server-Lösungen erreichbar, jedoch werden dabei die Verbindungen zwischen den einzelnen Diensten über den Client geschleift. Der Client muß die Daten zuerst von einem Dienst abholen um sie dann zu einem anderen Dienst weiterzuschicken. Bei Verbindungen zwischen zwei Diensten werden die Daten somit doppelt so oft übertragen als nötig. Bei Audiokommunikation kann das in heutigen LAN's noch tragbar sein. Bei Videodaten stößt man aber dann sehr schnell an die Grenzen der Bandbreite.

Unser System umgeht dies, indem die einzelnen Dienste direkt untereinander kommunizieren können, was jedoch eine ganze Reihe anderer Probleme aufwirft.

2 Grundanforderungen

Folgende Grundanforderungen sollten an das System gestellt werden:

- Weitestgehend grafisch ausgelegtes Servicedesign
  Für den Endanwender soll es möglich sein, nur mit grafischen Mitteln seine Applikation zu entwerfen. Dies betrifft im wesentlichen die Auswahl der Dienste und die Festlegung der Verbindungen der Dienste untereinander, so daß die gewünschte Funktionalität zustande kommt. Die Verwendung von speziellen Sprachen sollte vermieden werden und nur auf tiefer im System liegenden Schichten zur Anwendung gelangen.
- Netzwerktransparenz beim Zusammenstellen einer Applikation
  Die Information, auf welchem Rechner welche Funktionalität angeboten wird, dient nur als Ordnungskriterium bei der Auswahl durch den Endanwender. Für das weitere Vorgehen soll die Verteilung auf dem Netz transparent sein.

- Multiuser-Fähigkeit
  Die Teilkomponenten (im weiteren Services genannt) müssen so gestaltbar sein, daß eine ihrer Instanzen gleichzeitig von mehreren verschiedenen Benutzern verwendet werden kann. Als Beispiel stelle man sich einen Dienst vor, der das Ausgangssignal einer Videokamera an eine beliebige Zahl von Monitoren bei den Benutzern verschickt.
  Es soll nicht ausgeschlossen sein, daß mehrere Instanzen ein und derselben Teilkomponente gleichzeitig aktiv sind. Abspieldienste in einem (idealisierten) Video-On-Demand Server können hier als Beispiel genannt werden.
- Dynamische Services
  Ein Service muß so gestaltbar sein, daß er seine Struktur dynamisch zur Laufzeit ändern kann.
  Beim Beispiel der Konferenzschaltung könnte man dadurch die Zahl der Teilnehmer flexibel halten. Anfangs steht nur ein Anschluß für Teilnehmer zur Verfügung. Sobald dieser Anschluß belegt ist, wird vom Service ein neuer freier Anschluß bereitgestellt.
- Zumutbare Übertragungsqualität
  Bei der Übertragung von Audio- und Videodaten muß normalerweise die Verschlechterung des Signals in einem vernünftigen Rahmen bleiben. Im Rahmen dieses Projekts wird auf diesen Gesichtspunkt derzeit nur sehr rudimentär Rücksicht genommen. Es ist nicht Ziel dieses Projekts Lösungen für diese Problematik zu finden. Ein QOS-Management findet nicht statt.

3 Konzept

Das System wird als 3-stufige Client-Server-Architektur ausgelegt. Die Grundbausteine bilden sogenannte Services, die in der untersten Stufe untergebracht sind.

Sie sind autarke Untereinheiten, die über sogenannte Pins miteinander kommunizieren. Sie sind zum größten Teil als eigenständige Unix-Prozesse ausgeführt. Der Servicemanager in der zweiten Stufe hat die Aufgabe diese Services zu starten, Verbindungswünsche an sie weiterzugeben, und die Kommunikation zu den Clients (in der ersten Stufe) über das Netzwerk durchzuführen.

Die Funktionalitäten der einzelnen Stufen können auch teilweise kombiniert werden. So wird der Servicemanager zum Client, wenn er Compound-Services realisiert. In diesem Fall sieht es für die Clients so aus, als würde nur der Compound-Service existieren. Tatsächlich hat aber der Client-Teil eine ganze Reihe von Einzelservices angelegt und untereinander passend verbunden.

4 Strukturierte Verbindungen

Oft ergibt sich die Situation, daß für einen Service nicht von vornherein festgelegt werden kann oder soll, wieviele Anschlüsse vorzusehen sind. Als Beispiel sei hier ein Addierer für Audiosignale angeführt. Hier sind Multipins möglich, durch die ein Pin innerhalb der von der Hardware vorgegebenen Grenzen beliebig viele Subpins zur Laufzeit anlegen kann, die sich gleichartig verhalten aber innerhalb des Service noch unterscheidbar sind. Der Addierer hätte dann einen Multipin als Eingang und einen normalen Pin als Ausgang.

Um die Verkabelung der einzelnen Pins zu vereinfachen, wurden in Analogie zur Elektrotechnik Stecker implementiert, die mehrere Einzelpins zusammenfassen. Der in der Stufe 2 liegende Servicemanager sorgt dafür, daß alle Stecker-Verbindungen in einzelne Pin-Verbindungen aufgelöst und realisiert werden.

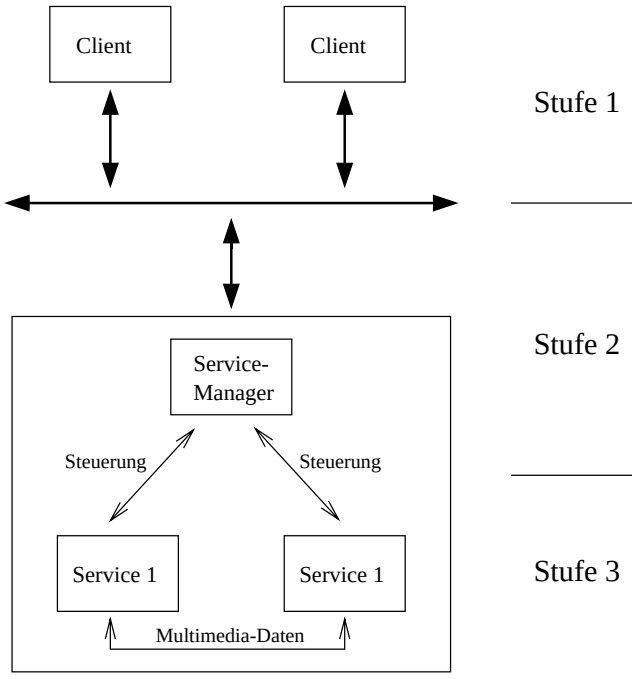

Fig. 1. 3-stufige Client-Server-Architektur

5 Primitive Services

Den Kern des Systems bilden die Primitive Services. Sie sind als Unix-Prozesse ausgelegt, die untereinander über Shared Memory, oder falls sie auf unterschiedlichen Workstations laufen, über TCP/IP verbunden sind.

Um das Erstellen von weiteren Services möglichst einfach zu gestalten, existiert eine Library, die alle Funktionen des Verbindungsauf- und -abbaus und der Kommunikation zum Servicemanager abwickelt. Der Autor eines solchen Services kann sich so auf die Implementierung der speziellen Funktion des Service konzentrieren. Er muß nur noch dafür sorgen, daß die Pins angelegt und mit den Multimediadaten versorgen werden, bzw. Methoden bereitstellen, die von den Pins aufgerufen werden sobald Daten von anderen Services eingetroffen sind.

Primitive Services übernehmen typischerweise den Zugriff auf die Multimedia-Hardware wie Mikrofon oder Kamera und führen die Signalverarbeitungsaufgaben wie Filterung, Spracherkennung usw. durch.

6 Interpreted Services

Interpreted Services sind eine Sonderform von Primitive Services, die als Dienst einen Interpreter bereitstellen. Das zu interpretierende File ist dann, sofern der Interpreter für verschiedene Plattformen vorliegt, plattformunabhängig und kann wenn nötig von einem Client zur Laufzeit bereitgestellt werden. Derzeit ist ein itcl-Interpreter realisiert, mit dem Bedienoberflächen in Services integriert werden können.

7 Compound Services

Compound Services können beliebige andere Services (auch Compound-Services) starten, verbinden und komplett als Gesamtservice dem Benutzer zur Verfügung stellen. Sie haben keinen direkten Kontakt zu den Daten, die zwischen Primitive Services ausgetauscht werden, sondern dienen nur der leichteren Organisierbarkeit.

Durch die Notwendigkeit auch Multigates und Multipins zur Verfügung zu stellen, muß ein umfangreiches Regelwerk berücksichtigt werden. Es muß die Zulässigkeit von Verbindungskombinationen geprüft werden und ggf. müssen Einzelkomponenten sinnvoll ergänzt oder entfernt werden.

8 Clients

Die Clients haben normalerweise (wenn sie nicht in Services eingebaut werden) keinen direkten Kontakt zu den Multimedia-Daten, sondern nur zu den Steuerinformationen für den Auf- und Abbau der Verbindungen zwischen den Services. Hierin unterscheiden sie sich von den herkömmlichen Client-Server-Systemen, die die Daten von einer Quelle zu einer Senke über den Client transportieren und dadurch das Netzwerk unnötig mehrfach belasten. Der derzeit einzige implementierte Client erlaubt die Überwachung der Server und das Starten und Verbinden von beliebigen Services mit Hilfe einer grafischen Bedienoberfläche. Durch diese allgemein gehaltene Funktion ist er relativ aufwendig. Im allgemeinen sind jedoch Clients mit wenigen Programmzeilen zu implementieren. Er ist derzeit unter Zuhilfenahme von tcl/tk realisiert und hat sich als sehr komfortables Hilfsmittel zum Austesten des Servers herausgestellt.

9 Status und Ausblick

Der Server ist mit Ausnahme der Compound-Services fertiggestellt. Services für Video- und Audio-Ein/Ausgaben sind in einer rudimentären Form verfügbar.

Derzeit wird daran gearbeitet den Client-Teil auch über WWW-Browser zugänglich zu machen. Der Schlüssel dazu liegt in der Sprache Java und dem in modernen Browsern integrierten Java-Interpreter. Die Clients brauchen dann keine für ihre jeweilige Hardwareplattform compilierte Version eines Programms vorrätig zu halten, sondern können das gewünschte Programm bei Bedarf in einer, dank Java hardwareunabhängigen Form, von einem HTTP-Server ihrer Wahl laden. Es ist noch unklar, wie Services auf dem WWW-Browser ihre Continous-Media-Daten präsentieren können. HTML unterstützt nur die Übertragung von Files, jedoch nicht das Öffnen von Kommunikationsverbindungen. Eine Lösung nach dem Vorbild von RealAudio ist denkbar, setzt aber zusätzlich zum Browser weitere Software voraus, was ja gerade durch den Einsatz von Java verhindert werden sollte.

An einer Komplettierung des Angebots an Services muß sicher noch gearbeitet werden.

Literaturverzeichnis

[Ang91] S. Angebranndt, R. L. Hyde, D. H. Luong, N. Siravara, C. Schmandt Integrating Audio and Telephony in a Distributed Workstation Environment, Usenix Summer '91 Nashville TN

[Aro92] B. Arons, Tools for Building Asynchronous Servers to Support Speech and Audio Applications, UIST '92 Monterey CA

[Dec93] DEC Cambridge Research Laboratory AudioFile System, Version 2, Release2, Release Notes and Installation Instructions, 1993