Technische Universität Berlin, Fakultät I, Fachgebiet
Kommunikationswissenschaft

Prof. Dr. Stefan Weinzierl

Exposé zur Magisterarbeit

„Aufbau, Betrieb und Optimierung ein Clusters von GNU/Linux Workstations für die Wellenfeldsynthese“

eingereicht von
Thilo Koch, <tiko@admin-box.com>

Berlin, 14. April 2008

1 Einleitung

Im Rahmen der Medienausstattung des Hörsaals H104 im Hauptgebäude der Technischen Universität Berlin wird ein Cluster von GNU/Linux-Workstations zur Steuerung eines Wellenfeldsynthesesystems (WFS), für einen 832-kanaligen Lautsprecherarray aufgebaut. Dabei muss sowohl die Hardwarekonfiguration als auch die Installation und Konfiguration der Softwarekomponenten spezifisch für den Einsatzzweck angepasst, bzw. zum Teil neu entwickelt werden.

Die Forschung zu verteilten Systemen und insbesondere High-Performance Clustern haben sich in den letzten Jahren stark entwickelt. Durch die Fortschritte von Hard- und Software in diesem Bereich ist es möglich geworden, einfache Workstations zu Clustern zu verbinden und so ein Vielfaches an Rechenleistung zu erhalten. Für die WFS-Anlage der TU Berlin im H104 wurde dieser Weg beschritten, um ein leistungsfähiges Cluster zum Audio-Echtzeit-Rendering bereitzustellen.

In der Arbeit wird untersucht, welche möglichen Cluster-Setups für Hard- und Software in Frage kommen und für diesen Zweck geeignet sind. Weiterhin sollen diese Setups auf ihre Ressourcennutzung hin untersucht und optimiert werden, um dem Benutzer die höchstmögliche Leistung bereitzustellen.

2 Aufgabenstellung

(1) Das Ziel dieser Arbeit ist es, ein solches – möglichst für ähnliche Systeme wieder verwendbares - Setup zu entwickeln, zu implementieren und zu dokumentieren.

Dabei geht es beim Aufbau des Clusters um grundlegende Setups für

- die Hardware und Hardwarekonfiguration,
- die Software zur Bereitstellung der Clusterinfrastruktur (Dienste, Monitoring, Tools),
- die Audio-, Renderingsoftware mit den Schnittstellen für Benutzer (Jack, Wonder).

(2) Nach der ersten Implementation des Systems sollen Möglichkeiten der Perfomance- und Serviceoptimierung der Clusterplattform untersucht werden, indem Ressourcenengpässe und Reserven des Systems analysiert werden.

(3) Daraus sollen konkrete Verbesserungsmaßnahmen entwickelt werden, die anschließend umgesetzt und getestet werden. Dabei werden insbesondere folgende Bereiche bearbeitet:

- I/O Performance,
- Preemptionslatenz des Betriebssystems,
- Multiprozessorimplementierung der Middleware (Betriebsystem und Audio-Server).

(4) Während der Erstellung der Arbeit sollen außerdem Fragestellungen und (Forschungs-) Perspektiven für zukünftige Entwicklungsmöglichkeiten und Alternativen sowohl des Clusters, als auch des WFS-Systems angegeben werden. Diesbezüglich sollen Aspekte der Erweiterung des Clusters, alternativer Rechnerarchitekturen (z.B. Cell-Architektur), sowie weitere Nutzungsmöglichkeiten (z.B. Supercollider) diskutiert werden.

Beim Aufbau von verteilten Systemen stellen sich verschiedene Probleme, die sich in dieser Form bei einem Einzelplatzrechner mit einem Prozessor nicht stellen [Coulouris2001], [Heiß2004].

- Heterogenität: Das verteilte System ist aus einer Vielzahl von verschiedenen Komponenten mit verschiedenen Funktionen aufgebaut – Audio- und IP-Netzwerk, Rendering-Knoten und Frontend-Workstation bzw. Client-Systemen, verschiedene Programmiersprachen. Entsprechende Kommunikationsprotokolle können diese konkreten Unterschiede verbergen, da sie von der konkreten Hard- und Software abstrahieren (z.B. TCP/IP-Stack, OSC-Protokoll). Außerdem dient die Middleware zur Vermittlung zwischen diesen Differenzen.
- Gleichzeitigkeit / Nebenläufigkeit: Im verteilten System treten Fälle auf, in denen Clients

gleichzeitig auf gleiche Ressourcen zugreifen wollen. Es sind spezielle Mechanismen notwendig, diesen gleichzeitigen Zugriff so zu steuern, daß die Daten konsistent bleiben. Dieses Problem tritt insbesondere bei verteilten Programmen (z.B. auf Mehrprozessorrechnern) auf, die auf gemeinsame Daten zugreifen. Das kann in den entsprechenden Programmen durch bestimmte Standardtechniken erreicht werden, z.B. Semaphore.

- Transparenz: Der Benutzer soll das System als Ganzes sehen und benutzen, anstatt als eine Anzahl verschiedener Komponenten. Verschiedene Aspekte des Systems werden so unsichtbar gemacht, dass die implementatorischen Aspekte wie Ort (der Daten bzw. der Programmausführung), gleichzeitiger Zugriff usw. verborgen werden. Dies geschieht insbesondere durch die „Middleware-Dienste“.

Weitere Problembereiche betreffen die Fehlerbehandlung, Skalierbarkeit und Sicherheit, die aber hier eine untergeordnete Rolle spielen.

Die genannten Aspekte haben auf den verschiedenen Ebenen des WFS-Systems unterschiedlich große Bedeutung, auf die in der Arbeit an entsprechender Stelle eingegangen wird.

2.1 Setup

Hardware

Die Grundstruktur des Clusters wird durch das Setup der Hardware festgelegt, entsprechend sind die oben genannten Aspekte für verteilte Systeme zu beachten - Verteilung des Audio-Stroms, Netzwerktopologie, Verteilung der Arbeitslast auf die Rechenknoten, Vermeidung von Ressourcenengpässen.

Die konkrete Hardware für das Projekt war gegeben. Die Diskussion der verwendeten Hardware soll dazu dienen, diejenigen Komponenten des Systems zu identifizieren, die für die Funktion sowie die Performance kritisch sind.

- Rechner: Boards, Speicher, CPU, Kompatibilität,
- Netzwerk: Adapter, Switches, Kabel,
- Audio: Soundcards, Switches.

Betriebssystem und Middleware

Die für die Administration und Wartung des Clusters notwendigen Funktionen müssen ermittelt und die entsprechenden Tools installiert und angepasst werden. Eventuell müssen für spezielle

Administrationsaufgaben Programme bzw. Scripte entwickelt werden. [Frisch1995]
Zu den notwendigen Funktionen gehören u.a.:

- vereinfachte Installation und Updates der Betriebssysteme,
- Dienste – wie Authentifizierung, automatisierte Konfiguration und Steuerung, Fileserver, Namensdienste
- Backup Management, Datensicherung und Firewall,
- Systemanalyse bzw. -überwachung (Monitoring),
- Wartung (Usermanagement, Mail, Software Updates usw.).

Diese Funktionen sollen durch quelloffene und lizenzkostenfreie Softwarekomponenten realisiert werden, um zum einen das Projektbudget zu schonen. Andererseits war das Ziel, größtmögliche Flexibilität im Umgang mit der Software zu haben (z.B. bei Modifikationen).

Audio- und Renderingssoftware

Als Audio- und Renderingsoftware kommt auf dem Cluster insbsondere Jack und Wonder zum Einsatz. Dabei stellt Jack [Davis2003] den Plugin-Host für die im Rahmen des Wonder-Projekts entwickelten Render-Plugins (tWonder) bereit. Des weiteren besteht das Wonder-System aus einer Anzahl von Modulen, Daemons und Client-Software. [Baalman2007]
Diese Software wird in dieser Arbeit im Hinblick auf ihre Integration in das Cluster und ihre Ressourcennutzung bzw. Performance untersucht.

2.2 Verbesserung von Performance und Service

Der grundlegende Aufbau des Clusters geschieht zuerst mit verfügbaren Standard-Workstation Komponenten. Dadurch wird es überhaupt erst möglich mit vertretbarem Aufwand ein so komplexes System wie das WFS-System aufzubauen. Im Verlauf des Aufbaus sowie des (Test-) Betriebs können dann die spezifischen Bedingungen der konkreten Anwendung ermittelt werden, um das System danach weiter zu optimieren.
Aufgrund der Komplexität des Systems ist dieses “inkrementelle” Vorgehen (Aufbau – Test – Verbesserung) eine Möglichkeit, den sonst sehr aufwändigen Planungs- und Entwicklungsprozess im Vorfeld der Installation zu reduzieren und auf Änderungen der Anforderungen oder des Aufbaus flexibel reagieren zu können. Außerdem ist eine genaue theoretische Vorausbestimmung der Betriebsbedingungen und Eigenschaften des Systems in der Planungsphase nicht möglich.
Des weiteren schafft eine solche Vorgehensweise automatisch Strukturen und Tools, die es vereinfachen, das Clustersystem weiter zu entwickeln.
Ein solches Vorgehen ist bei sehr komplexen Softwarevorhaben oft zu finden, z.B. Linux Kernel Entwicklung, Xtreme Programming [Wake2001].

Zur Optimierung von Performance und Service sind jeweils folgende Schritte notwendig:

- Messung ausgewählter Betriebsparameter,
- Interpretation der erhaltenen Messwerte,
- Entwicklung und Implementierung von Änderungen,
- Vergleichsmessungen bzw. Tests zur Erfolgsüberprüfung.

Eventuell sind mehrere Iterationnen dieses Ablaufs notwendig.
Die voraussichtlich zu untersuchenden Bereiche für Optimierung sind der Betriebssystemkernel, die I/O-Operationen und die Audiosoftware.

Betriebssystemkernel (linux)

Der Kernel des Betriebsystems verwaltet die (Rechen-)Prozesse, die Daten und alle weiteren Ressourcen (Rechenzeit, Speichernutzung, I/O-Operationen). Die dabei eingesetzten Algorithmen und Verfahren bestimmen insbesondere die Laufzeiteigenschaften der verschiedenen Prozesse (Laufzeit, Latenz, Throughput, Realtime-Qualität).

Neben der für das Rendering notwendigen Rechenleistung spielt insbesondere die Preemptionslatenz eine wichtige Rolle, da nur bei geringen Latenzen eine unterbrechungsfreie – knackserlose – Audiowiedergabe möglich ist. Obwohl der eingesetzte Kernel (Linux) keinen echten Realtime Modus (Hard RT) besitzt, kann der Kernel so modifiziert werden, daß er zumindest sehr nahe an die Eigenschaften eines RT-Systems herankommt – so genanntes Soft RT [Corbet2004].
Der Linux-Kernel des Betriebsystems besitzt vielfältige Möglichkeiten der Konfiguration und somit der Anpassung an spezielle Anforderungen durch das WFS-System. Zusätzlich existieren eine Reihe von experimentellen Patches zur Verbesserung der Echtzeitfähigkeit. [RealTimeWiki]

Die Wirksamkeit verschiedener Konfigurationen und Patches soll getestet werden.

I/O-Operationen

Zur Speicherung von Programmen und Daten verwenden die Workstations Festplatten. Dabei können verschiedene Filesysteme benutzt werden. Aufgrund ihrer verschiedenen Eigenschaften sind sie unterschiedlich gut für das Clustersystem und insbesondere die Audio-Anwendung (RT Forderung) geeignet. Es sollen verschiedene Filesysteme getestet werden.

Die Cluster-Knoten kommunizieren zum größten Teil über ein IP-Netzwerk miteinander (Filesystem, BS-Dienste, OSC-Nachrichten). Obwohl die Netzwerkperformance nicht im engeren Sinne performancekritisch ist, sind doch unnötige Latenzen und Ressourcenengpässe zu vermeiden. [Tanenbaum2003]

Optimierung der Audiosoftware (Mehrprozessorfähigkeit)

Das verwendete Setup setzt auf das Rendering in einem verteilten System. Dabei wird Arbeitslast zum einen auf verschiedene Workstations verteilt. Auf der anderen Seite stehen in jeder Workstation zwei Prozessoren bereit. Die Verteilung der Arbeitslast auf die Prozessoren kann nach verschiedenen Methoden geschehen. In der Arbeit soll die optimale Verteilung (Scheduling) der Arbeitslast im Hinblick auf die Anforderungen unseres Systems und unter Berücksichtigung der unter Punkt 2 genannten Problematiken ermittelt werden.

Der in unserem System eingesetzte Audio-Host ist das „Jack-Audio-Connection-Kit“ (Jack). In seiner derzeitigen Implementation ist es nicht multithreadfähig und kann somit die Dual-Core Prozessoren der Workstations nicht auslasten. Im Rahmen dieser Arbeit soll Jack durch eine weiter entwickelte – multiprozessorfähige – Version (Jackdmp) ersetzt werden [Letz2005]. Entsprechend müssen die Renderplugins des Wonder Projekts angepasst und eine sinnvolle Scheduling Strategie implementiert werden.

2.3 Dokumentation

Für den Rechner-Cluster werden - entsprechend den verschiedenen Nutzerperspektiven - verschiedene Dokumentationen bzw. Handbücher benötigt. Folgende Perspektiven sind vorgesehen:

- Cluster Administration und Wartung,
- Cluster Benutzung für die WFS,
- Developer Dokumentation für die entwickelten Funktionen.

Diese Dokumentation soll Online verfügbar gemacht werden (z.B. in einem Wiki).

3 Quellen

[Frisch1995]	Frisch, Aleen.: Essential System Administration, O'Reilly & Associates, 1995
[Coulouris2001]	Coulouris, George; Dollimore, Jean; Kindberg, Tim: Distributed Systems, Addison-Wesley, Pearson Education, Harlow, 2001
[Tanenbaum2003]	Tanenbaum, Andrew. S.: Computer Networks, Pearson Educational, New Jersey, 2003
[Wake2001]	Wake, William C.: Extreme Programming Explored, Addison Wesley, Pearson Education, Harlow, 2001

Linkografie

[Heiß2004]	Heiß, Hans-Ulrich: Vorlesung: Betriebssysteme in verteilten Umgebungen,

Skript, Technische Universität Berlin, http://kbs.cs.tu-berlin.de/teaching/sose2004/bsvum/bsvum_index.htm, Zugriff am 14.04.2008.

[Baalman2007] Baalman, Marije: WFS in electronic music, http://www.kgw.tu-berlin.de/~baalman/program, Zugriff am 14.04.2008

[Davis2003] Davis, Paul: The Jack Audio Connection Kit, Vortrag auf Linux Audio Developers Conference, Karslruhe, 2003, http://lad.linuxaudio.org/events/2003_zkm/slides/paul_davis-jack/title.html, Zugriff am 14.04.2008

[Letz2005] Letz, Stephane; Fober, Dominique; Orlarey, Yann: jackdmp: Jack server for multi-processor machines, in Linux Audio Conference 2005 Proceedings, Köln, 2005, http://lac.zkm.de/2005/papers/lac2005_proceedings.pdf, Zugriff am 14.04.2008.

[Corbet2004] Corbet, Jonathan: Approaches to realtime Linux, http://lwn.net/Articles/106010, Zugriff am 14.04.2008.

Low Latency Howto, http://lowlatency.linuxaudio.org

[RT-wiki] Real-Time Linux Wiki, http://rt.wiki.kernel.org/index.php/Main_Page

[FrontRunner] FrontRunner Computer Performance Consulting: Understanding Computer Performance Analysis, http://www.frontrunnercpc.com/info/index.htm, 2002

[comp.benchmarks] comp.benchmarks FAQ, http://pages.cs.wisc.edu/~thomas/comp.benchmarks.FAQ.html

4 Arbeits- und Zeitplan

Aufgabe	*Aufwand*
Softwarerecherche und Evaluation	2 Wochen
Installation der Cluster- , Administrationssoftware und Tools	2 Wochen
Tests	von Beginn an begleitend
Verbesserung von Performance und Service	
- Recherche und Evaluation	1 Wochen
- Installation und Test	3 Wochen
Dokumentation und schriftliche Arbeit	4 Wochen

5 Gliederungsentwurf

1 Einleitung

2 Aufbau des Clusters

2.1 Hardware

2.2 Betriebssystem und Middleware

2.3 Audio- und Renderingsoftware

3 Betrieb des Clusters

3.1 Administration (Server, Dienste, Backup, Netzwerk, Monitoring)

3.2 Scripte und Tools

3.3 User Perspektive

4 Fortentwicklung und Optimierung

4.1 Messung des aktuellen Systems

4.1.1 I/O-Operationen (Netzwerk, Festplatten)

4.1.2 Preemptionslatenz und Echtzeitbedingung

4.2 Schlussfolgerungen und Verbesserungen

4.3 Implementierung und Messung

4.4 Mehrprozessorfähigkeit

4.4.1 Theorie

4.4.2 Stand der Technik: jackd -> jackdmp

4.4.3 twonder -> twondermp

4.4.4 Implementierung und Messung

5 Diskussion und Ausblick