BERUFSAKADEMIE MANNHEIM
STAATLICHE STUDIENAKADEMIE

Fachrichtung Informationstechnik

Referat Seminar Informatinostachnik

Thin-Client Architekturen

Ingo Scholz

Doreen Wetzel

Inhaltsverzeichnis

1. Rechnerarchitekturen

1.1. Terminal / Großrechner

Bei der Terminal / Großrechner Architektur greifen die Benutzer über Terminals auf einen Großrechner (englische Bezeichnung: Host oder Mainframe) zu. Alle Anwendungen sind auf diesem Großrechner installiert und werden von dort ausgeführt. Die Terminals sind an der Ausführung der Anwendungen nicht beteiligt und dienen nur der Dateneingabe und – ausgabe. Die Datenausgabe erfolgt über einen einfachen Bildschirm (meist so genannte „Green Screen"- textterminals), für die Dateneingabe ist eine Tastatur angeschlossen.

Abbildung 1: *Dieses Bild zeigt einen typischen Mainframe*

Im Allgemeinen besitzt ein Terminal keine lokalen Datenträger (so z.B. keine Festplatten, Diskettenlaufwerke) und kein lokales Betriebssystem. Ein Terminal bootet über das Netzwerk vom Host. In der so genannten Firmware sind nur die wichtigsten Tastaturbefehle zum Großrechner schicken können. Ein Terminal ist rein textbasiert, das heißt, es können keine Grafiken dargestellt werden. Somit gibt es keine graphische Bedienoberfläche.

Abbildung 2: *Terminal*

Die ersten Terminal/Host Systeme gab es etwa seit Mitte der 60er Jahre (z.B. *IBM System/360* von 1964). Anfänglich waren sie noch mit Lochkarten, Magnetbändern und einfachen Druckern für die Datenein- und Datenausgabe ausgestattet, dann folgten die ersten textbasierten Terminals. Zwischen Host und Terminals bestand bei IBM eine Koaxial-Verbindung, später wurde eine Twisted Pair Verkabelung eingesetzt. Zur Kommunikation dienten zunächst proprietäre Protokolle, wie z.B. IBMs SNA Architektur. Im Laufe der Zeit kamen Protokolle wie Telnet (ab 1969) und TCP/IP (in kommerzieller Großrechnerumgebung ab Anfang der 90er Jahre) zum Einsatz.
Terminal/Host-Architekturen sind klar strukturierte Systeme (siehe Abbildung /3/). Es sind nur die auf dem Host installierten Applikationen verfügbar und der Zugriff auf diese kann über Benutzerrechte geregelt werden.

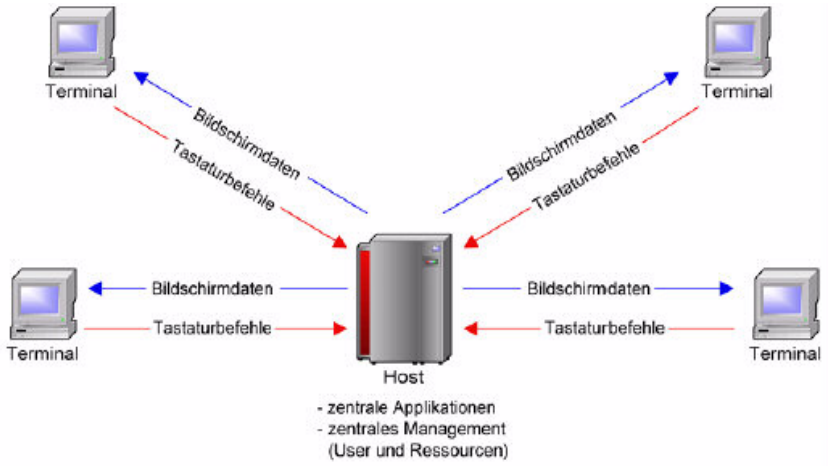

Abbildung 3: *Terminal/Host-Architektur*

Vorteile:

- Zentrales System-Management möglich
  Dazu zählen unter anderem zentrale Softwarebereitstellung, Benutzerverwaltung, etc.
- Hohe Zuverlässigkeit
  Der Großrechner trägt alle Last und ist aufgrund seiner Wichtigkeit in der Regel mehrfach abgesichert. Sollten Terminals ausfallen (was aufgrund ihrer einfachen Ausstattung unwahrscheinlich ist), können diese schnell und unkompliziert ausgetauscht werden, ohne das System als Ganzes zu berühren.
- Leistungsfähigkeit der Endgeräte ist unerheblich

Nachteile:

- Proprietäre Systeme
- Keine Standardanwendungen (unflexibel, da spezialisiert)
- Kostenintensiv in Anschaffung und Unterhalt (Großrechner)
- weniger intuitive Handhabung für den Benutzer (keine GUI)

	Terminal	Host
Applikationen	Keine	Alle benötigten Applikationen sind auf dem Host installiert und werden dort auch ausgeführt
Betriebssystem	Keins	Proprietäres Host-Betriebs-system, z.B. IBM Z/OS (OS/390) oder Fujitsu-Siemens BS2000/OSD
Hardware	Firmwarechip oder -board, Bildschirm und Tastatur	Großrechner-Hardware der Hersteller

1.2. Client/Server

Mit der Entwicklung der ersten leistungsfähigen Einzelplatz-PCs ab 1981 entstand das Bedürfnis die Komplexität und Leistungsfähigkeit von den in der Regel teuren Großrechnersystemen auf verteilte Arbeitsplatzrechner zu verlagern. Der Rechner wurde unabhängiger von der Verfügbarkeit und Performance des zentralen Systems. Deshalb spricht man in diesem Zusammenhang auch von Dezentralisierung, da die Daten- und Anwendungsverarbeitung vor Ort, d. h. lokal auf den Workstations stattfinden kann. Das traditionelle Client/Server-Modell wurde geboren. Der Kern des Client/Server-Modells besteht aus den beiden Funktionen *Client* und *Server* (vergleiche Abbildung /4/). Der Client agiert als Dienstnachfrager, der einen Dienst10 beim Server in Anspruch nehmen will. Dementsprechend dient der Server, der seine Dienstbereitschaft zuvor signalisiert hat, als Dienstanbieter. In der Regel kommunizieren Client und Server über ein Netzwerk miteinander. Ein Client kann auf verschiedene Server zugreifen und ein Server kann mehrere Clients bedienen.

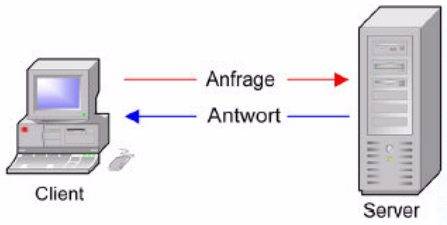

Abbildung 4: *Client/Server - Architektur*

Unter Umständen können die Rollen (Funktion als Client oder Server) auch wechseln. Benötigt ein Server beispielsweise den Dienst eines weiteren Servers, tritt er diesem gegenüber als Client auf (siehe Abbildung /5/).

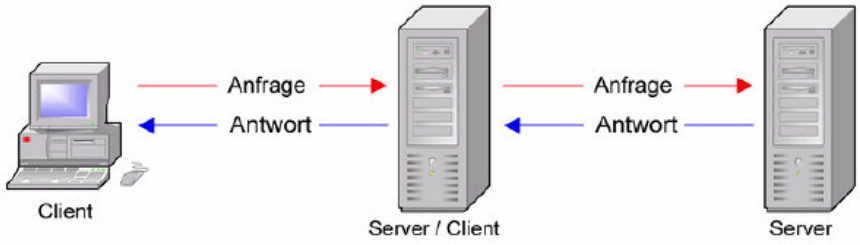

Abbildung 5: *Client/Server - Doppelrolle*

Es gibt verschiedene Ausprägungen des Client/Server-Modells in Bezug auf die Schichtenverteilung (Präsentations-, Anwendungs- und Datenschicht). Prinzipiell lassen sich Client/Server-Architekturen in Zwei-Schichten, Drei-Schichten oder N-Schichten Modelle aufteilen (auch Two-Tier, Three-Tier oder Multi-Tier genannt), abhängig von der Art der Verteilung der Schichten auf den Institutionen. Bei der Zwei-Schichten Architektur mit den beiden Institutionen Client und Server gibt es fünf verschiedene Modelle (siehe Abbildung /6/):

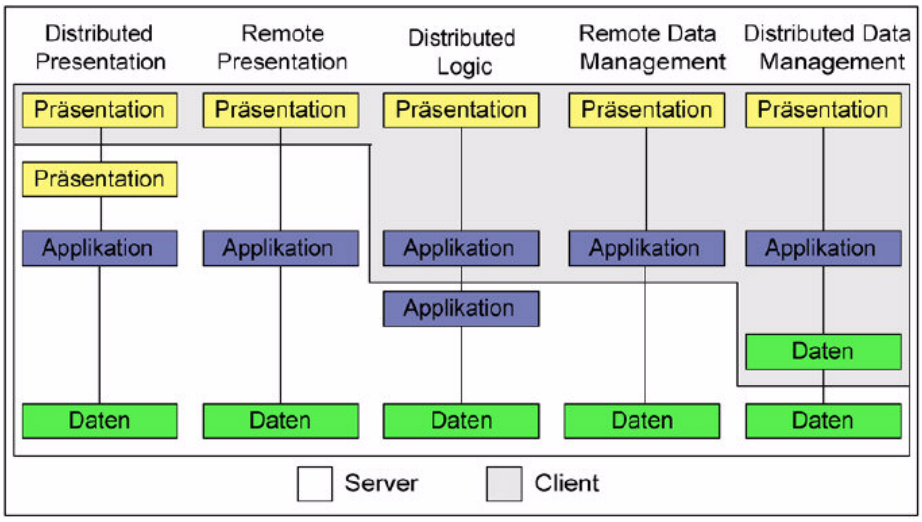

Abbildung 6: *Client/Server – 2-Schichten-Modell*

- *Distributed Presentation*

Die Präsentation der Anwendung (User-Interface) findet auf dem Client statt. Der Client stellt lediglich ein Terminal dar. Die gesamte Verarbeitung, inklusive der Aufbereitung der Daten für die Präsentation, übernimmt der Server.

Beispiele für dieses Verteilungsmodell sind grafische User-Interfaces (GUI), wie z.B. zeichenorientierte Terminalemulationen.

- *Remote Presentation*

Dem Client ist die vollständige Darstellungsschicht zugeordnet. Der Server hält alle Anwendungen und Daten für den Client vor. Als Beispiel ist ein Web Browser zu nennen, der HTML-Daten vom Web-Server erhält und diese selbständig darstellt.

- *Distributed Logic*

Beim Modell der Distributed *Logic* teilen sich der Client und der Server die Anwendungsschicht. Die Präsentationsschicht ist vollständig dem Client und die Datenschicht vollständig dem Server zugeordnet. Für die Funktionsfähigkeit ist es nötig, dass die voneinander getrennten Anwendungselemente miteinander kommunizieren. Die Implementierung eines solchen Modells ist dementsprechend komplex.

- *Remote Data Management*

Die Präsentations- und Anwendungslogik befindet sich auf dem Client (Fat-Client). Der Server stellt den Clients die Daten, die er selbst vorhält, zur Verfügung. In der Praxis werden meist Datenbank-Systeme eingesetzt.

In der Regel werden Standard Büro-Anwendungen, wie z.B. Textverarbeitung und E-Mail-Programme innerhalb dieser Architektur verwendet.

- *Distributed Data Management*

Beim *Distributed Data Management* sind die Präsentations- und Anwendungslogik, sowie ein Teil der Datenlogik dem Client zugeordnet. D.h., die Daten liegen sowohl auf dem Client wie auch Server.

Aufgrund dieser Verteilung muss ein Mechanismus vorhanden sein, der die Datenkonsistenz garantiert. Bei der Drei-Schichten-Architektur kommt im Vergleich zum Zwei-Schichten-Modell noch eine weitere Ebene hinzu, auf die die Schichten aufgeteilt werden. Im Allgemeinen handelt es sich dabei unter anderem um einen File- oder Datenbank- Server (siehe Abbildung /7/).

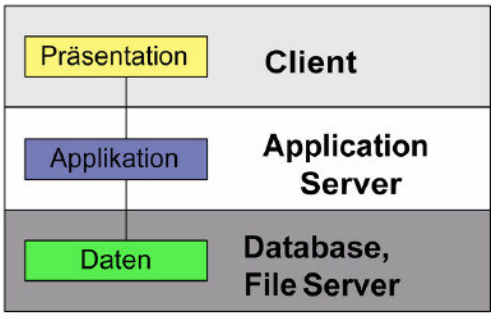

Abbildung 7: *Client/Server – 3-Schichten-Modell*

Analog dazu verhält es sich bei einer N-Schichten-Architektur, d.h. die Schichten werden zusätzlich auf mehrere Ebenen verteilt. Beim klassischen Client/Server-Modell handelt es sich um eine Zwei-Schichten Architektur, genauer gesagt um eine Remote Data Management Architektur. Das bedeutet, dass der Client vollständig für die Präsentations- und Anwendungslogik zuständig ist und sich der Server nur um die Bereitstellung der
Daten bzw. Ressourcen kümmern muss. Die heutzutage am meisten eingesetzte Software basiert auf diesem Modell.
Aufgrund der Anforderung an den Client, sich sowohl um die Präsentationsschicht (d.h. die Darstellung der Anwendungen) als auch um die Anwendungsschicht (d.h. die lokale Ausführung der Anwendungen) kümmern zu müssen, ist es deshalb für diese Architektur notwendig, dass der Client mit ausreichend leistungsstarker
Hardware ausgestattet ist. Grundvoraussetzung für diese Architektur ist auch die Bereitstellung von ausreichender Bandbreite für das Netzwerk. Dieses Modell ist gegenwärtig als Standard (da sehr weit verbreitet) anzusehen. Die Kommunikation zwischen Client und Server wird über verschiedene Standard-Protokolle geregelt. Dazu zählen unter anderem TCP/IP, IPX/SPX (von Novell), NetBIOS (von IBM), NetBEUI (von Microsoft) oder Appletalk (von Apple-Macintosh).

Im Vergleich zu Großrechnersystemen bietet die Client/Server-Architektur eine Reihe
an Vorteilen:

- Unabhängigkeit vom Server, da bedingt selbständiger (d.h. autonomer) und rechenstarker Client
- Durch die Leistungsfähigkeit des Clients, sind lokal auch rechenintensive Anwendungen möglich (z.B. CAD)
- Herstellerunabhängigkeit
- große Auswahl an Standard-Anwendungen, z.B. Büro-Anwendungen wie Textverarbeitung, o.ä.

- große Hardware-Auswahl
- geringere Hard- und Softwarekosten, da Standards verwendet werden können
- Flexibilität
- Skalierbarkeit ist horizontal (Aufrüstung der PCs) als auch vertikal (Aufrüstung der Server) möglich
- Ergonomie der Benutzeroberfläche von PCs und Workstations (GUI)

Der Benutzer muss nicht mehr mit textbasierten, einfarbigen Anzeigen arbeiten. Mehrfarbige, fensterorientierte Benutzeroberflächen sind benutzerfreundlicher.

Dennoch sind auch gewisse Nachteile nicht zu vernachlässigen, die einer besonderen Beachtung bedürfen:

- hohe Komplexität durch Verteilung
- komplexes Management der Datenkonsistenz, Datensicherheit, etc.
- Gesamtkosten schwer überschaubar
- hohe Netzbelastung
- teure Systemwartung
- kurze Innovationszyklen (Hard- und Software)
- meist sehr verschiedene Ausstattungen (keine homogene Endgeräte-Landschaft)

	Client	Server
Applikationen	Alle Anwendungen laufen auf dem Client	Beim klassischen Client/Server Modell keine Anwendungen
Betriebssystem	Client-Betriebssystem, z.B. Windows 2000, Windows NT Workstation, Linux	Standard Server Betriebs-systeme, z.B. Windows NT Server, Unix (HP-UX, IBM AIX, etc.)
Hardware	Voll-Ausstattung: CPU, RAM, Grafikkarte, Festplatte, Diskettenlaufwerk, Bildschirm, Tastatur, Maus	Server-Hardware, die speziell auf die Anforderungen aus-gelegt ist. Beim klassischen Client/Server Modell Standard-Komponenten und vor allem große Festplattenkapazität

13. Server-based Computing

Unter Server-based Computing (SBC) versteht man eine spezielle Client-Server- Architektur, bei der die Anwendungen zu 100 Prozent auf dem Server12 installiert, verwaltet und ausgeführt werden. Der entfernte Client dient sozusagen nur als Anzeige- und Eingabemedium. Dieses Modell ist daher nicht mit der klassischen Client/Server-Architektur zu verwechseln. Beim Server-based Computing Modell wird die Applikationslogik von der Benutzerschnittstelle abgekoppelt. Der Client dient nur noch zur Darstellung der Daten. Wie in Abbildung /8/ ersichtlich, werden zwischen Server und Client nur die Tastaturbefehle, Mouse-Aktionen und Bildschirm-aktualisierungen, sowie Audio über das Netzwerk übertragen.

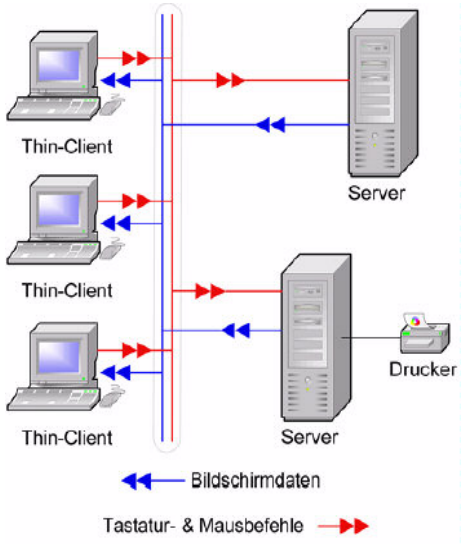

Abbildung 8: *Server-based Computing*

Aufgrund dieser definierten Funktionen kann die Ausstattung des Clients auf ein Minimum reduziert werden. Dieses Konzept erinnert an die Terminal/Host-Architektur. Tatsächlich verbindet das Server-based Computing die Vorteile des Terminal/Host-Prinzips (leistungsstarker Server, einfache Clients) mit den Vorteilen des Client/Server-Prinzips (Skalierbarkeit und benutzerfreundliches GUI). Server-based Computing lässt sich nach dem 2- oder 3-Schichten-Konzept realisieren. Der Server kann als zweite Instanz (zum Client) die Anwendungs- und Datenbereitstellung übernehmen (2-Schichten-Modell, siehe Abbildung /9/) oder ein Server dient nur als Applikationsserver und die Schicht der Datenhaltung wird auf eine dritte Instanz verteilt (3-Schichten-Modell, siehe Abbildung /10/).

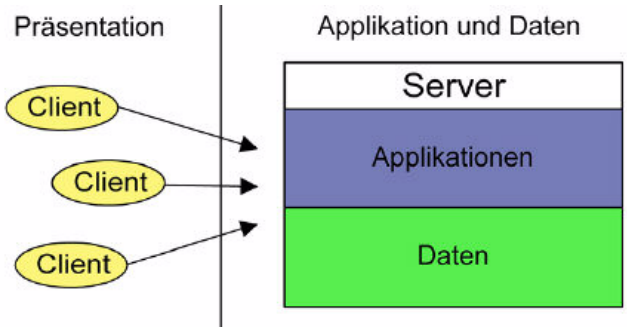

***Abbildung* 9:** *Server-based Computing 2-Schichten-Modell*

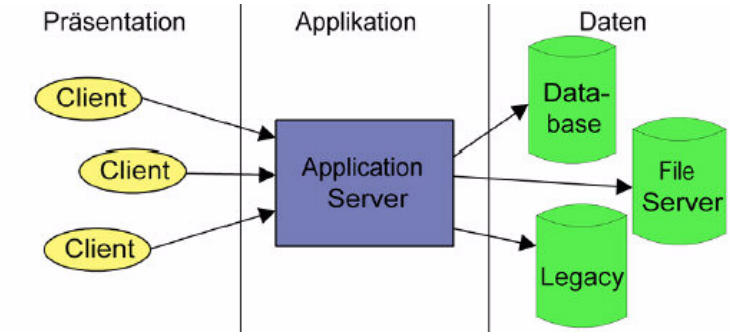

Abbildung 10: *Server-based Computing 3-Schichten-Modell*

Eine zentrale Komponente des Server-based Computing ist in der Regel auch eine umfangreiche Managementfunktion. Der zentrale Server sollte mit einer Management-Software ausgestattet sein, die es ihm erlaubt, die angeschlossenen Clients und die installierte Software effektiv zu verwalten. Server-based Computing lässt sich in der Regel innerhalb der bereits bestehenden Infrastruktur umsetzen und arbeitet mit gängigen IT-Standards, sowie Standard- Anwendungen. Die Server-based Computing Architektur wird oft auch Thin-Client/Server Computing oder ASP (Application Service Provider) Environment genannt.

2. Thin-Client Architekturen

Thin-Client Architekturen basieren auf dem Server-based Computing-Konzept. D.h., alle Daten und die Software liegen auf dem Server und die Applikationen werden dort auch ausgeführt. Die Clients hingegen dienen nur der Darstellung. Thin-Client Architekturen verbinden die Vorteile der Terminalverwaltung mit den Vorzügen einer Umgebung, die mit benutzerfreundlichen GUIs und Standardanwendungen arbeiten kann.
Eine Thin-Client Architektur besteht prinzipiell immer aus den Komponenten Server und (Thin-)Client, wobei jede Komponente wiederum Hard- und Software-Aspekte beinhaltet. Natürlich müssen diese beiden Komponenten auch miteinander kommunizieren; dies erfolgt über definierte Protokolle, die von der Server-Software abhängig sind.

2.1. Die Server-Komponente

Der Server innerhalb einer Thin-Client Umgebung stellt den Thin-Clients alle erforderlichen Daten und Anwendungen zur Verfügung.
Da die Thin-Clients alleine nicht funktionsfähig sind, spielt der Server die wichtigste Rolle. Deshalb werden an den Server hohe Anforderungen in Bezug auf seine Hard- und Software, sowie Verfügbarkeit gestellt werden. Aufgrund der entsprechenden Ausstattung werden die Server innerhalb einer Thin-Client Umgebung oft auch Fat-Server genannt.
Die Hardware eines Fat-Servers muss so leistungsfähig sein, dass sie den Anforderungen eines Applikations- und Datenservers bezüglich Prozessorkapazität, Arbeitsspeicher, Festplattenspeicher, etc. genügt. Dabei spielen beim Grad der Ausstattung mehrere Faktoren eine Rolle: unter anderem, ob der Server als Applikations- und/oder Datenserver agiert, wieviele Thin-Clients bedient werden müssen, welche Anwendungen (gleichzeitig) ausgeführt werden müssen, usw.
Prinzipiell sind die Hardwareanforderungen an einen Fat Server in einer Thin-Client Architektur sehr hoch. So kommen meistens Serversysteme oder Serververbunde (Serverfarm) mit mehreren Prozessoren und Arbeitsspeicher im Gigabyte-Bereich zum Einsatz.
Die Server-Software besteht unter anderem aus dem zwingend multiuserfähigen Server-Betriebssystem und einer speziellen Kommunikations-Software, die die benötigten Anwendungen für die Thin-Clients bereitstellt (siehe Abbildung /11/).

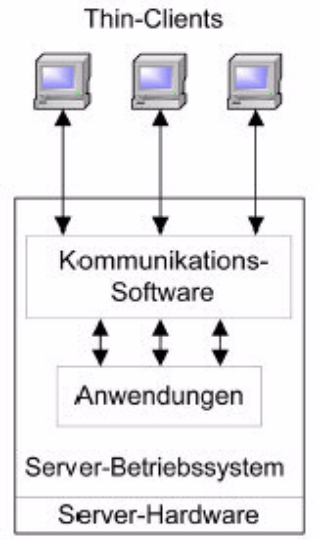

Abbildung 11: *Server Komponenten*

Sollen mehrere Benutzer gleichzeitig Zugriff auf den Server erhalten, ist zwingend ein Mehrbenutzerbetrieb, bzw. Multiuser-Betrieb notwendig. Der Multiuser-Betrieb enthält stets die Multitasking-Fähigkeit, da beim Zugriff mehrerer Benutzer natürlich zwangsläufig mehrere Anwendungen zur Ausführung kommen.

Die Kommunikations-Software, die zentraler Bestandteil der Thin-Client Architektur ist, macht es möglich die Anwendungslogik von der Präsentationsschicht zu trennen, so dass der Thin-Client an der Ausführung von Anwendungen nicht beteiligt ist und sich nur mit dem Darstellen der Anwendung beschäftigen muss. Die Kommunikations-Software vermittelt dem Client nur noch die nötigen Informationen. Zur Kommunikation gehört das Empfangen der Tastatur- und Mausbefehle des Thin-Clients und das Zurücksenden der aktualisierten Bildschirmdaten sowie gegebenenfalls Audiosignale. Dies geschieht dabei über ein gemeinsames Protokoll, welches Display- oder Kommunikationsprotokoll genannt wird. Diese speziellen Protokolle sind herstellerabhängig, d.h. eine bestimmte Kommunikations-Software zieht ein bestimmtes Display-Protokoll nach sich.

Der tatsächliche Transport der Kommunikations-Daten erfolgt in der Regel über Standard-Transport-Protokolle wie zum Beispiel TCP/IP oder UDP. Die Display-Protokolle werden in den Transport-Protokollen eingebettet.

Die Server-Software erhält von der Anwendung die entstandenen neuen Informationen, bereitet diese in Form von Darstellungsdaten auf und schickt sie eingebettet im Display- bzw. Kommunikationsprotokoll an den Thin-Client.

Die spezielle Kommunikationssoftware muss mit dem Betriebssystem des Servers kompatibel sein und auf dem entfernten Thin-Clients muss ein entsprechendes Gegenstück zur Kommunikation mit der Server-Software installiert sein.

2.2. Die Client-Komponente

Der Client innerhalb einer Thin-Client Architektur wird aufgrund seiner geringen Ausstattung (Soft- und Hardware), seiner fest vorgegebenen Aufgabe und auch seiner Größe wegen Thin-Client genannt. Er benötigt zwingend den Server und muss mit ihm kommunizieren, denn alleine ist der Thin-Client nicht funktionsfähig.
Da der Sinn einer Thin-Client Architektur darin liegt, alle Intelligenz und Komplexität vom Client weg hin zum Server zu verlagern, braucht ein Thin-Client prinzipiell nur eine Grundausstattung, die ihm folgendes ermöglichen muss:

- Aufbau einer Verbindung zum Server
- Empfangen der Audio- und Displaydaten, sowie Senden der Tastatur- und Mausbefehle
- Auswerten der empfangenen Daten und Ausgabe über Lautsprecher bzw. Bildschirm

Thin-Clients bieten dem Benutzer eine gewohnte grafische Benutzeroberfläche (GUI) und können mittlerweile (je nach gewähltem Hersteller bzw. Modell) mit Windows-, Unix und Mainframe-Anwendungen arbeiten und über einen Browser mit
Internetanwendungen umgehen.
Es existieren mehrere Philosophien zur praktischen Umsetzung auf dem Markt, so dass es eine Reihe unterschiedlicher Konzepte bzw. Ausprägungen gibt, wie Thin-Client-Architekturen auf der Client-Seite realisiert werden können. Je näher ein Thin-Client (anhand seiner Ausstattungsmerkmale) einem voll ausgestatteten PC kommt, desto dicker, d.h. „fatter“ wird er (siehe Abbildung /12/).

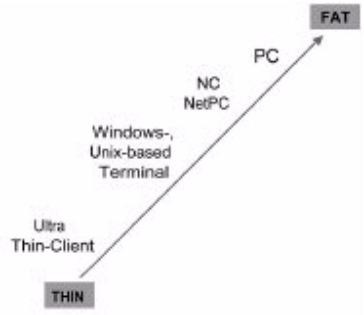

Abbildung 12: *Thin-Client Spektrum*

2.2.3. PCs mit Thin-Client Anwendung

Die „fattest“ Art das Thin-Client-Konzept umzusetzen, ist die Verwendung eines herkömmlichen PCs mit seiner üblichen Ausstattung, auf dem man eine so genannte Thin-Client Anwendung laufen lässt (PC als intelligenter Terminal). Diese Form des Thin-Clients wird oft auch als Hybrid-PC bezeichnet.
Ein PC besteht prinzipiell aus folgenden Hardware-Komponenten:

Der PCist ein selbständiger und leistungsfähiger Computer. Aufgrund stetiger Weiterentwicklungen in Bereich der Software müssen die PCs natürlich nachziehen, um diese neu entwickelten Anwendungen performant ausführen zu können. Daraus folgt zwingend ein Hardware-Upgrade.

Wird ein PC aber als Thin-Client genutzt, können unter Umständen diese aufwendigen und in der Regel teuren Upgrades entfallen – wenn das Thin-Client Konzept konsequent umgesetzt wird. Nahezu jeder PC (auch ältere Modelle) kann mit einer installierten Thin-Client Anwendung und mit Zugriff auf einen entsprechenden Server zu einem Thin-Client Endgerät umfunktioniert werden. So kann auf bereits existierende Hardware zurückgegriffen werden und eine Neuanschaffung von Thin-Client Endgeräten kann entfallen bzw. mengenreduziert werden.
Wie Abbildung /13/ zeigt, kann die Umstellung dabei alleine über eine besondere Anwendung geschehen, die auf dem Client einfach installiert wird oder über den hardwareseitigen Einbau einer so genannten Thin-Client Einsteckkarte (mit der Thin-Client Software „on Chip“).

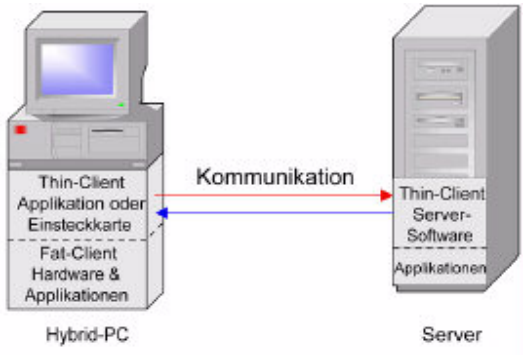

Abbildung 13: *Hybrid-PC*

Vorteile:

- Da es sich im Prinzip immer noch um einen voll ausgestatteten PC handelt (Fat-Client), können lokal rechenintensive Anwendungen (z.B. CAD) ausgeführtwerden.
  Ein Fat Client erfüllt in der Regel alle Anwendungsanforderungen (sowohl Textverarbeitung als auch CAD-Anwendung).
- Bei Bedarf kann der Client auch autonom, d.h. unabhängig vom Server arbeiten, muss dabei aber auf lokale Anwendungen zurückgreifen.
- Der Benutzer arbeitet weiterhin in seiner bekannten Umgebung, damit ist dieses Thin-Client-Konzept sehr bedienerfreundlich.

- Da auf vorhandene, alte Hardware zurückgegriffen wird, kann dieses Thin-Client-Konzept kostensparend umgesetzt werden.
- Zudem lässt sich die erforderliche Thin-Client-Anwendung einfach installieren und der Thin-Client ist schnell verfügbar.

Nachteile:

Aufgrund der Verwendung eines herkömmlichen PCs, existieren weiterhin die typischen Schwachstellen eines Fat-Clients:

- Hardware-Fehleranfälligkeit durch viele bewegliche Teile (z.B. Festplatte, Lüfter, etc.). Das bedeutet, die MTBF bleibt unverändert niedrig.
- Der Benutzer behält weiterhin gewisse Handlungsfreiräume gegenüber dem Umgang mit dem PC. Viele Sicherheitsaspekte sind nicht zu vernachlässigen.
  Dazu zählen unter anderem Datensicherheit, Virengefahr, Installation nicht genehmigter Software.
- Sollten lokale Anwendungen erhöhte Anforderungen an die Hardwareausstattung des Hybrid-PCs stellen, könnte ein Hardware-Upgrade nötig werden und damit das Thin-Client Prinzip untergraben.

2.2.4. Netzwerkcomputer

Unter den Begriff Netzwerkcomputer fallen die beiden Lösungen Networkcomputer (NC) und NetPC. Eigentlich gehören diese Lösungen nicht zu dem Thin-Client Konzept, da sie dieses bei näherer Betrachtung aus verschiedenen Gründen nicht erfüllen. Dazu zählen unter anderem die Art der Kommunikation, die bewusst lokale Ausführung von Anwendungen und das Dulden lokaler Speichermedien. Innerhalb der Literatur und des Internets werden diese Systeme häufig doch als Thin-Client bezeichnet werden.

NC:

Das NC-Konzept wurde gemeinsam von Apple, Sun, IBM, Netscape und Oracle entwickelt und 1995 der Öffentlichkeit vorgestellt. Wie der Name Networkcomputer schon vermuten lässt, benötigt der NC zwingend eine Netzwerkverbindung: Software und auch das Betriebssystem werden über das Netz (auch über das Internet) von einem zentralen Server auf den NC geladen und lokal ausgeführt (siehe Abbildung / 12 /). Deshalb ist der NC mit einem schnellen Prozessor und ausreichend Arbeitsspeicher ausgestattet.

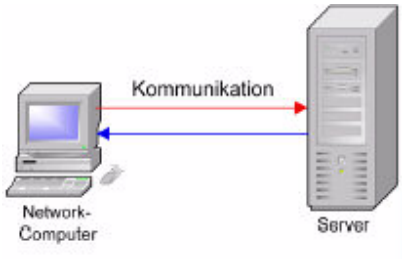

Abbildung 12: *Network-Computer*

Der Networkcomputer basiert auf der plattformunabhängigen Java-Technologie; er arbeitet primär mit *Java-Applets* innerhalb eines Java-Browsers oder mit Java-Anwendungen in der *Java Virtual Machine* (JVM).
In der Regel besitzt der NC keine lokalen Speichermedien, sondern bezieht seine Daten nur über das Netzwerk und legt sie auch dort wieder ab. Der NC besteht aus einem verschlossenen Gehäuse, an dem ein Bildschirm, eine Tastatur und eine Maus angeschlossen sind und das eine Netzwerkschnittstelle besitzt.
Da der NC seine Software und Daten lokal verarbeitet, ist er dem PC näher als einem Terminal.
Obwohl das Konzept durchaus seine Berechtigung hatte, wurde der NC nicht wirklich angenommen. Unter anderem scheuten sich die Anwender vor dem Zugriff über das Internet, bzw. das Ablegen ihrer Daten an einem ihnen unbekannten entfernten Ort.

NetPC:

Aus der auf Java basierenden NC-Idee heraus, quasi als Gegenzug, entwickelten Microsoft und Intel ein eigenes Konzept.
Dieser NetPC hat fast die gleichen Eigenschaften wie ein herkömmlicher PC. Er besitzt einen leistungsstarken Prozessor und lokale Speichermedien. Die Anwendungsverarbeitung findet ausschließlich lokal statt. Der NetPC ist über ein Netzwerk mit einem Server verbunden, um von dort aus gemanagt zu werden und auch Internet-Konnektivität zu erhalten. Das entsprechende Management-Tool von Microsoft heißt *Zero Administration for Windows (ZAW)*.
Aufgrund seiner starken Prozessorleistung und den lokalen Speichermedien ist der NetPC „fatter“ als der NC. Der NetPC konnte sich ebenfalls nicht durchsetzen.

2.2.5. Windows- bzw. Unix-based Terminals

Windows- und Unix-based Terminals (WBTs bzw. UBTs) sind Terminals der neuen Generation. Sie sind in der Regel über ein Netzwerk an einen Windows Terminal Application Server oder Unix Server angeschlossen und ermöglichen den Zugriff auf 16- und 32-bit Windows-, Java-, Unix- und Mainframe-Anwendungen (siehe Abbildung / 13 /).

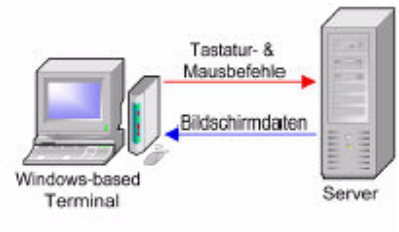

Abbildung 13: *Windows-based Terminals*

Zudem bieten sie eine benutzerfreundliche grafische Anwendungsoberfläche (GUI). Das angepasste Betriebssystem (Windows CE embedded, Windows CE.NET, Windows XP embedded, Windows NT

embedded oder Linux embedded), sowie eine minimale Client-Software (für die Verbindung zum Server notwendig) sind optimalerweise in der so genannten Firmware in Form eines Chips (meist EEPROMFlash) gespeichert.
Auf dem Markt werden aber mittlerweile verschiedene Ansichten und Konzepte über WBT bzw. UBT vertreten, so dass man ganz genau differenzieren muss. Die Ausstattung der Windows- und Unix-based Terminals variiert stark. In der Regel besitzen sie keine beweglichen Teile, wie zum Beispiel Festplatten, Lüfter usw. und es werden lokal keine Anwendungen ausgeführt. Es gibt aber Ausnahmen, bei denen ein WBT bzw. UBT lokal Daten ablegen oder auf lokale Peripheriegeräte (z.B. CD-ROM, Scanner, etc.) zugreifen kann und auch spezielle Anwendungen wie z.B. Browser oder Emulationen (für den Zugriff auf einen am Netz angeschlossenen Host) lokal ausgeführt werden, womit solche WBT und UBT „fatter“ werden und näher an den PC rücken.
Vorteile:

- in der Regel sehr einfache Hardware-Ausstattung
- keine Hardware-Upgrades nötig
- meist keine beweglichen, fehleranfälligen Teile wie z.B. Festplatten oder Lüfter, damit erhöht sich die MTBF
- meist keine Wärmeentwicklung
- meist sehr leise
- geringerer Energieverbrauch als herkömmlicher PC
- niedrigere Anschaffungskosten
- schnelle Installation des Endgerätes (meist Plug 'n' Play)
- defekte Geräte lassen sich schnell und leicht austauschen
- höhere Datensicherheit, wenn lokal keine Anwendungen laufen und keine lokalen Speichermedien angeschlossen sind (Virenschutz, Datenschutz)
- benutzerfreundlich
- zentrales Management möglich

Nachteile:

- sehr verschiedene Konzepte, genauere Untersuchung notwendig
- Serverabhängigkeit

2.2.6. Ultra Thin Clients

Der reinste Thin-Client realisiert das Server-based Computing am konsequentesten. Tatsächlich kommt diese Art von Thin-Client ohne jegliche lokale Anwendungsverarbeitung aus und besitzt definitiv keine beweglichen Teile. Aufgrund dieser strikt reduzierten Ausstattung ohne Ausnahmen heißt dieses Endgerät Ultra Thin-Client. Ein Ultra Thin-Client ist konsequent nur mit den aller nötigsten Komponenten ausgestattet, um als Anzeigemedium zu fungieren. Dazu zählen:

- ausreichender Chipsatz zur Bedienung der Ein- und Ausgabefunktionen
- leistungsfähiger Grafikchip zur schnellen Darstellung der Inhalte in guter Auflösung und Bildwiederholrate, Schnittstellen für Tastatur, Maus, Monitor und Netzwerk (Netzwerkkarte), ausreichend Arbeitsspeicher für das idealerweise in einem Chip untergebrachte Basisbetriebssystem. Alles auf einem Board in einem kompakten Gehäuse. Sie brauchen keine weiteren Schnittstellen.

Dieser Client lässt sich natürlich auch mit Peripheriegeräten ausstatten, aber an dieser Stelle wird aus dem Ultra Thin-Client ein WBT. Bei den Ultra Thin-Clients ist die Intelligenz tatsächlich zum Server verlagert. Alle Leistung liegt beim Server, der daher eine überdurchschnittlich hohe Leistungsfähigkeit besitzen muss.

Vorteile:

- minimalste Hardware-Ausstattung
- überhaupt keine beweglichen, d.h. mechanisch hoch beanspruchte Teile
- Langlebigkeit
- sehr geringe Wärmeentwicklung
- sehr leise
- minimaler Energieverbrauch
- sehr leichtes und kleines Endgerät
- hohe Datensicherheit in Bezug auf Datenschutz und Virenbefall durch Anwender
- kostengünstig
- die in der Regel hohe Leistungsfähigkeit des Servers erlaubt die Ausführung von Multimedia-Anwendungen.

Nachteile:

- Sehr eng mit den Fähigkeiten des Servers verbunden, da lokal überhaupt keine Anwendungen oder Betriebssysteme laufen
- keine aufwendigen Anwendungen wie CAD möglich

2.3. Anwendungsbeispiele der Thin-Client-Architektur

Thin-Client-Architekturen sind mit einer Reihe von Vorteilen verbunden. Anhand dieser Vorteile lassen sich potentielle Anwendungsgebiete ableiten. Es bietet sich an, Thin-Client-Konzepte dort umsetzen, wo viele Anwender auf die gleichen Applikationen zugreifen, unter Umständen sogar an wechselnden Orten. Abgesehen davon eignen sich auch die Bereiche, in denen mit Terminals gearbeitet wird und die auf ein benutzerfreundlicheres grafisches User-Interface umsteigen wollen. Die Leistungsfähigkeit wird auf einen zentralen Server verlagert, so dass am Anwenderarbeitsplatz kostengünstige Endgeräte eingesetzt werden können, ohne dass die Anwendungsperformance darunter leiden muss. Durch eine zentrale Instanz lassen sich die Systemkomponenten einfach und effizient managen:

- Die Software wird auf dem Server ausgeführt, eine Installation vor Ort entfällt,
- Benutzer erhalten nur auf bestimmte Anwendungen Zugriff,
- Der früherer „Vor-Ort-Support“ kann nun zentral über ein Session-Shadowing erfolgen,
- Die Thin-Client-Endgeräte sind alle gleich. Bei einer Fehlfunktion bzw. bei einem Defekt wird das entsprechende Gerät einfach durch ein neues ersetzt und der Anwender kann ohne nennenswerte Unterbrechung mit seiner Arbeit fortfahren.

Die Thin-Client-Architektur lässt sich am sinnvollsten in Umgebungen einsetzen, in denen gleichartige Funktionen verwendet werden. Daher sind folgende Anwendungsszenarien denkbar:

- Bildungswesen (Schulen, Hochschulen, Bibliotheken, etc.) Schulen und Hochschulen bieten den Schülern bzw. Studenten in der Regel Möglichkeiten im Internet zu surfen oder ihre Hausarbeiten an den schul- bzw. hochschuleigenen PCs zu verfassen. Zudem erhalten Schüler oft in eigenen Kursen eine Grundschulung hauptsächlich in Standardanwendungen wie Microsoft Word, u.ä. Auch in Bibliotheken stehen zur umfassenden Literatur-Recherche PCs mit Anschluss an einen Server bzw. das Internet bereit.

  Gerade im Bildungswesen lässt sich daher das Thin-Client-Konzept sehr gut umsetzten. Eine große Anzahl an gleichartigen Thin Clients kann über einen zentralen Server auf alle Anwendungen zugreifen, ohne dass diese lokal installiert werden müssen. Über den Server könnten dann auch zusätzliche Datenbanken erreicht werden. In klassichen PC-Umfeld neigen die Anwender, d.h. die Schüler, häufig dazu unautorisierte Software zu installieren oder das System umzukonfigurieren. Da die Thin-Clients (bis auf das PC-Modell) ohne Serveranbindung funktionsunfähig sind, verringert sich auch die Diebstahlgefahr. Im Falle eines defekten Gerätes, lässt sich dieses schnell und unkompliziert (auch von der Lehrkräften) gegen ein Neues austauschen. Das zentrale Management ist hier von besonderer Bedeutung.
- Industrie, Transportwesen, Banken, Behörden, Gesundheitswesen Auch in der Industrie gibt es Bereiche, in denen der Einsatz von Thin-Clients durchaus gerechtfertigt ist. Überall dort, wo Systeme nur mit definierten Aufgaben betraut sind oder PCs nur dazu dienen, auf mehreren Bildschirmen die gleichen Daten darzustellen.

In Bahnhöfen und Flughäfen existieren im allgemeinen sehr viele Darstellungsmedien (Anzeigetafeln, Monitore, etc.). Sogar die Informations-, Selbstbedienungs- bzw. Ticketterminals für die Kunden könnten mit der Thin- Client-Technologie ausgestattet werden. Auch im Bereich der Banken, Behörden, im Gesundheitswesen und im Handel generell (Geschäftsfilialen) können Thin-Clients eingesetzt werden. Dort greifen viele Mitarbeiter ständig auf unternehmensweite Datenbanken- und Anwendungssysteme zu, wobei sie örtlich nicht an die Endgeräte gebunden sein dürfen (Bankschalter, Call Center, Kassen, usw.). Im Geschäftsumfeld können normale Office-PCs, die nur mit den Standardanwendungen wie z.B. Textverarbeitung, Tabellenkalkulation, Browser und Email- Client arbeiten, aus funktionaler Sicht durch Thin-Clients ersetzt werden. Nur für Anwender, die mit Anwendungen arbeiten, die sehr hohe Leistungsansprüche besitzen (z.B. CAD), sind PCs weiterhin die einzig sinnvolle Lösung.

Literaturverzeichnis

/1/ Larisch, Dirk Das Einsteigerseminar Netzwerktechnik, vmi, 2001

/2/ Lipinzki, Klaus Lexikon der Datenkommunikation, DATACOM Buchverlag GmbH

/3/ Mathers, T. W. Windows NT / 2000 Thin Client Solutions

/4/ Kaplan, Steve Server-Based Computing, mitp-Verlag, Landsberg,2001
Mangus, Marc

/5/ http://www.golem.de

/6/ http://www.igel.de

/7/ http://www.wyse.de/sc-computing/which-tc.html

/8/ http://www.nue.et-inf.uni-siegen.de/~schmidt/tcsecurity/over.html

/9/ http://de.sun.com/Produkte/Hardware/Server/index.html

/10/ http://h41113.www4.hp.com/thinkthin/de/ger/

/11/ Datenblatt HP Compaq Thin Clients