Linux Solutions Group e.V.
Breitscheidstraße 4
70174 Stuttgart
http://www.lisog.org/
info@lisog.org

LiSoG»Szenario Thin Client
Arbeitspapier

Prototyp eines Open Source basierten Thin Clients

Nico Gulden
Linux Solutions Group e.V.

Mentor:
Horst Bräuner
Stadtverwaltung Schwäbisch Hall

September 2006

Impressum

Autor

Nico Gulden, Linux Solutions Group e.V.

Mitwirkende

Horst Bräuner, Stadtverwaltung Schwäbisch Hall
Dr. Karl-Heinz Strassemeyer, Vorsitzender LiSoG
Klaus Haasis, Geschäftsführer LiSoG
Karl-Eugen Binder, Vorstand LiSoG
Rainer Bessert, Fujitsu Siemens Computers ITPS

Erstellt mit OpenOffice.org 2.0
Satz: LaTeX

Herausgeber

Linux Solutions Group (LiSoG) e.V.
Geschäftsstelle Stuttgart
Breitscheidstr. 4
70174 Stuttgart
`http://www.lisog.org/`
E-Mail: `mailto:info@lisog.org`

Creative Commons - Commons Deeds

Namensnennung - Keine Bearbeitung 2.0 Deutschland

Sie dürfen:
den Inhalt vervielfältigen, verbreiten und öffentlich aufführen den Inhalt kommerziell nutzen.

Zu den folgenden Bedingungen:
Namensnennung. Sie müssen den Namen des Autors/Rechtsinhabers nennen. Keine Bearbeitung. Der Inhalt darf nicht bearbeitet oder in anderer Weise verändert werden.
Im Falle einer Verbreitung müssen Sie anderen die Lizenzbedingungen, unter die dieser Inhalt fällt, mitteilen. Jede dieser Bedingungen kann nach schriftlicher Einwilligung des Rechtsinhabers aufgehoben werden.

Die gesetzlichen Schranken des Urheberrechts bleiben hiervon unberührt. Das Commons Deed ist eine Zusammenfassung des Lizenzvertrags in allgemeinverständlicher Sprache.

Laut einer Studie von IDC von 2005 bleibt der Thin Client Markt auch in Zukunft eine Wachstumsbranche[1]. Die Vorteile von Thin Clients liegen auf der Hand. Zentrales Anwendungssoftware-Management, längere Nutzungsdauer, höhere Zuverlässigkeit und ein günstigerer Preis im Vergleich zu PCs sprechen für den Einsatz von Thin Clients und Server-based-Computing.

Fokus des LiSoG-Szenarios Thin Client sind linuxbasierte Thin Clients. Obwohl es eine Vielzahl von Herstellern für linuxbasierte Thin Clients gibt, existieren keine reinen Open Source und linuxbasierten Thin Clients und kein einheitliches Thin Client Management. Im Linux und Open Source Bereich sieht die LiSoG einen Bedarf an dieser Form vom Computerarbeitsplätzen und widmet sich der Beschreibung eines linuxbasierten Open Source Thin Clients, um Machbarkeit und Handlungsbedarf aufzuzeigen.

Dieses Dokument beschreibt einen Prototyp für einen solchen Thin Client. Es erläutert die Ausgangslage und die Zielsetzung des Szenarios. Es gibt Einblick in die allgemeine Architektur einer Thin Client Umgebung und beschreibt den Softwarestack des linuxbasierten Open Source Thin Clients. Ein Ausblick nennt weiterführende Ideen und noch zu erarbeitende Bereiche wie USB over IP oder USB/SIM Authentisierung.

Der Thin Client Prototyp der Stadt Schwäbisch Hall zeigt, dass die Bestandteile für einen linuxbasierten Open Source Thin Client vorhanden sind. Linux und das X Window System bieten hervorragende technische Voraussetzungen für eine Thin Client Umgebung. Wirtschaftlich ermöglicht eine Thin Client Umgebung die weitere Nutzung ausgedienter PCs, indem sie zu Thin Clients umfunktioniert werden. Durch die NX Technologie erfährt der Anwender keine Einschränkungen in der Performance der Anwendungen gegenüber einem PC.

Der Prototyp der Stadt Schwäbisch Hall zeigt jedoch weitere Aufgaben auf. So fehlt es beispielsweise an einem Hersteller oder IT Dienstleister, der Thin Clients mit den Software Images ausliefert und auch Wartung und Support für die Geräte anbietet. Des Weiteren fehlt es an einer zentralen Systems Management Software Lösung für Thin Clients, die verschiedene Thin Clients auch über Distributionsgrenzen hinweg verwaltet. Auch technisch bietet ein Open Source Thin Client mit USB over IP oder USB/SIM Authentisierung weitere Herausforderungen, die in diesem Dokument beschrieben werde.

[1] Quelle: `http://www.etcf.de/cms/presse_meldungen_mitglieder_2005_de/detail.php?nr=5058&kategorie=presse_meldungen_mitglieder_2005_de`

Inhaltsverzeichnis

1. Über die LiSoG **2**
- 1.1. Vorstellung der LiSoG . . . 2
- 1.2. Szenarien & Solution Stacks . . . 2
- 1.3. Open Source . . . 2

2. Motivation und Zielsetzung **3**

3. Prototyp der Stadt Schwäbisch Hall **4**
- 3.1. X Window System . . . 4
- 3.2. Die NX Technologie . . . 6
- 3.3. Terminalserver . . . 7
- 3.4. Server-Backend . . . 8
- 3.5. Thin Client . . . 8
- 3.6. Thin Client Software . . . 11

4. Anzugehende Problemlösungen **14**
- 4.1. Benchmarking . . . 14
- 4.2. Hard- und Software . . . 14
  - 4.2.1. Hersteller für Standard Thin Client gesucht . . . 14
  - 4.2.2. Systems Management für Thin Clients gesucht . . . 15
- 4.3. Technische Implementierungen . . . 15
  - 4.3.1. USB over IP . . . 15
  - 4.3.2. USB/SIM Authentisierung . . . 16

5. Fazit **18**

Literaturverzeichnis **19**

A. Hersteller **20**
- A.1. Thin Client Hardware . . . 20
- A.2. Thin Client Software . . . 21
- A.3. Berichte . . . 21

1. Über die LiSoG

1.1. Vorstellung der LiSoG

Die Linux Solutions Group e.V. (LiSoG) wurde im März 2005 mit dem Ziel gegründet, linuxbasiertes Business im deutschsprachigen Raum zu fördern. Durch die aktive Einbindung und Vernetzung der Mitglieder wie IT-Anwender und -Anbieter, sowie Unternehmen des öffentlichen Sektors und wissenschaftliche Institutionen will die LiSoG zur Erhöhung der Marktakzeptanz und des Vertrauens für Gesamtlösungen auf Linux und Open Source beitragen.

1.2. Szenarien & Solution Stacks

Ein Arbeitsschwerpunkt sind Szenarien & Solution Stacks. LiSoG-Szenarien umfassen konkrete Themenbereiche, die sich mit Anwenderszenarien befassen. Sie werden anwender- und lösungsorientiert betrachtet und zeigen auf, welche linuxbasierten Lösungen auf dem Markt existieren. Anhand von Anforderungen werden vorhandene Lösungen analysiert und beschrieben. Die Lösungen nutzen hierbei das Open Source Betriebssystem GNU/Linux und bilden einen Solution Stack.

1.3. Open Source

Der Begriff Open Source beschreibt die Freiheit eines Einzelnen, Einblick in den Quelltext von Software zu nehmen. Hierbei handelt es sich um einen Marketing-Begriff, der „Freie Software als geschäftsfreundlich und weniger ideologisch belastet darstellen“ soll. Er wurde von Eric S. Raymond, Bruce Perens und Tim O'Reilly geprägt und gab der Open Source Initiative ihren Namen. Der Begriff Freie Software ist auf Richard Stallman zurück zu führen. Es gibt verschiedene Open Source Lizenzen. Die bekannteste ist die GNU General Public License (GPL). Bekannte Software unter der GPL sind beispielsweise der Betriebssystemkern von Linux, KDE, OpenOffice.org und Blender. Trotz der Vielfalt der Open Source Lizenzen, weißen sie die folgenden drei charakteristischen Merkmale auf[1]:

- Der Quellcode einer Software liegt in einer für den Menschen lesbaren und verständlichen Form vor.
- Die Software darf beliebig kopiert, verbreitet und genutzt werden.
- Die Software darf verändert und in der veränderten Form weitergegeben werden.

Neben der GPL seien noch Lesser GNU General Public License (LGPL), BSD License, Apache License und Mozilla Public License als weitere bekannte Open Source Lizenzen genannt.

[1] Artikel *Open Source*. In Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 16. Januar 2006 01:39 UTC URL: `http://de.wikipedia.org/wiki/Open_source` (Abgerufen: 13. Februar 2006)

2. Motivation und Zielsetzung

Die Rückbesinnung zur Zentralisierung der Administration von Computern in Richtung Server-based-Computing macht Fat-Clients in vielen Abteilungen von Firmen, Kommunen und Organisationen unnötig. Es kommen zunehmend Pläne auf, PC-basierte Rechnerumgebungen durch Thin Clients zu ersetzen. So strebt beispielsweise die Stadt Schwäbisch Hall eine Umstellung ihrer Desktop-PCs zu Thin Clients in Teilen ihrer IT-Infrastruktur an. Auch die Stadt Freiburg erwägt im Zuge der eigenen Hardwarestandardisierungsstrategie den Einsatz von Thin Clients als einfache Systeme für Server-based-Computing. Das Land Berlin evaluiert ebenso ThinClient-Konzepte bezüglich Ergonomie und Wirtschaftlichkeit. Einige hundert Benutzer werden dort bereits durch Terminalserver bedient.

Die Vorteile der zentralen Benutzer- und Softwareadministration liegen auf der Hand. Hinzu kommen der niedrige Stromverbrauch, der einfache Austausch der Geräte, geringe Wärmeentwicklung und die niedrige bis keine Geräuschentwicklung.

Durch die architektonische Trennung von Server- und Client-Funktionen ist GNU/Linux besonders für Thin Client Umgebungen geeignet.

Für den Betrieb einer Thin Client Umgebung sind Server für die Ausführung der Anwendungen und Thin Clients notwendig. Der Thin Client spaltet sich in Hardware- und Software-Komponente auf. Die Hardware-Ausstattung richtet sich nach den Anforderungen des Anwenders und wird in diesem Szenario vorerst nicht weiter berücksichtigt. Der Fokus des Szenarios richtet sich auf die Software-Komponente des Thin Clients.

Das LiSoG-Szenario Thin Client möchte einen komplett linuxbasierten Open Source Thin Client beschreiben. Es befasst sich vorrangig mit einem aus Standard-Komponenten bestehenden Software-Stack auf der Thin Client Hardware. Dieser Software-Stack soll leicht paketierbar und für die Massenproduktion geeignet sein.

Der Prototyp der Stadt Schwäbisch Hall des Open Source basierten Thin Clients bietet einen aktuellen Linuxkernel mit entsprechenden aktuellen Softwarepaketen. Damit wird ein einheitliches hardwarehersteller-unabhängiges Software-Image geschaffen, das leichter in bestehende Administrationsinfrastrukturen integrierbar ist.

Grundlage hierfür soll ein Linux-Betriebssystem in Minimalausführung sein, dass ausschließlich den Betrieb der Thin Client Hardware und die Kommunikation zum Server unterstützt. Mit dem Prototyp der Stadt Schwäbisch Hall ist eine effiziente Thin Client Umgebung implementiert worden. Sie wird in diesem Dokument im Detail beschrieben. Darauf aufbauend werden Zukunftstechnologien diskutiert.

3. Prototyp der Stadt Schwäbisch Hall

Ein durchschnittlicher Computerarbeitsplatz in einem Unternehmen stellt dem Mitarbeitern einen PC zur Verfügung. Dieser PC verfügt über die notwendige Software, damit der Mitarbeiter seine tägliche Arbeit erledigen kann. Dazu zählen mindestens ein E-Mailprogramm, ein Webbrowser, eine Office-Anwendung und andere Software. Die Daten, wie beispielsweise Office-Dokumente, liegen hierbei meist zentral auf einem Server. Dies gilt prinzipiell für Windows- und Linux-Arbeitsplätze.

Architektonisch gesehen verfügt der Linux-Arbeitsplatz über eine Besonderheit. Für die grafische Darstellung von Anwendungen, wie Webbrowser und E-Mailprogramm, kommt das X Window System zum Einsatz.

3.1. X Window System

„Das X Window System (auch: X Version 11, X11, X, aber nicht X-Windows) ist eine Sammlung von Protokollen, Computerprogrammen und Standards zur Ansteuerung grafischer Bildschirme im Allgemeinen und zur Anzeige einer grafischen Benutzungsoberfläche, vor allem unter Unix-Systemen.“[1] Um die Darstellung des Webbrowser-Fensters kümmert sich dieses X Window System. Es basiert auf dem Client-Server-Modell bestehend aus X-Server und X-Client und ist netzwerktransparent. Abbildung 3.1 zeigt schematisch einen Linux-Arbeitsplatzrechner, der über X-Client und X-Server verfügt.

Der X-Server kommuniziert mit den Geräten für die Ein- und Ausgabe, zum Beispiel Maus, Tastatur, Grafikkarte und Bildschirm. Über eine rastergrafikbasierte Darstellung erfolgt das Zeichnen von Linien und Mustern. Der X-Server ist zuständig für die ereignisorientierte Handhabung von Maus oder Tastatur. Unter Unix-Systemen ist der X-Server die Grundvorraussetzung zur Darstellung von grafischen Anwendungen und stellt somit ein Minimalsystem zur Verfügung. XFree86 und X.Org sind Implementierungen des X-Servers [Wik06].

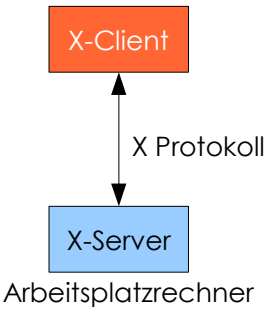

Abbildung 3.1.: Linux Arbeitsplatzrechner

Jedes grafische Anwendungsprogramm, wie beispielsweise der Webbrowser Mozilla Firefox, ist ein X-Client. Der X-Server übernimmt die Darstellung des X-Clients und reicht ihm verschiedene Ereignisse wie Tastatureingaben und Mausbewegungen weiter.

Abbildung 3.2 zeigt das Client-Server-Modell des X Window Systems. Die Kommunikation über die gestrichelte Linie in der Abbildung läuft netzwerktransparent über das X Protokoll ab. X-Server und X-Client können auf demselben Computer ausgeführt werden, wie in Abbildung 3.2. Das ist bei Desktop Systemen der Fall und die häufigste Anwendung. Wird auf einem PC beispielsweise ein Linux mit einem KDE oder GNOME Desktop installiert, so befinden sich X-Server und die X-Clients auf einem PC. Die X-Clients sprechen den lokalen X-Server an und dieser übernimmt die Darstellung.

[1] Artikel *X Window System*. In Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 19. März 10:43 UTC URL: `http://de.wikipedia.org/wiki/X-Window` (Abgerufen: 21. März 2006)

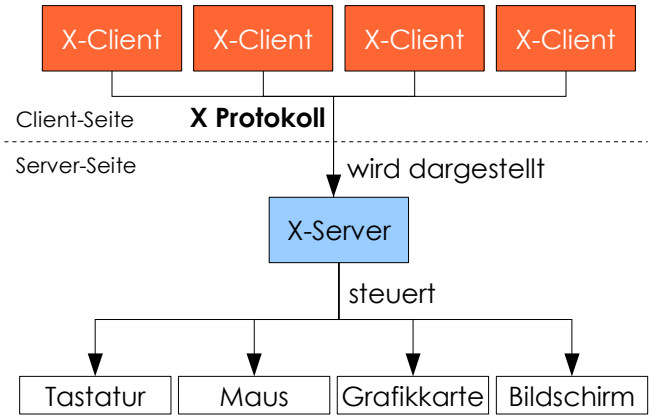

Abbildung 3.2.: X Window System

Durch das Client-Server-Modell ist es dem X Window System außerdem möglich beide Komponenten auf unterschiedlichen Systemen zu betreiben. So können die Anwendungen in Form der X-Clients zentral auf ein System ausgelagert und verwaltet werden und deren Darstellung findet dezentral auf dem X-Server des Arbeitsplatzsystems statt, wie Abbildung 3.3 zeigt.

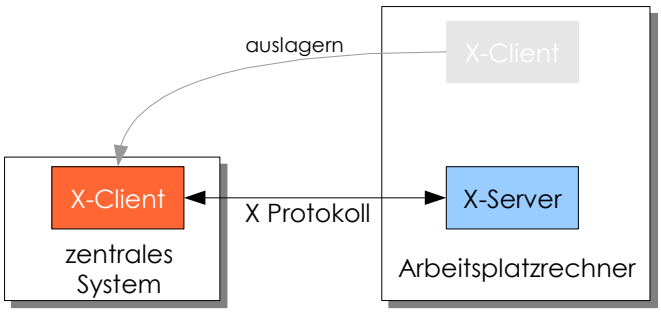

Abbildung 3.3.: Physische Trennung von X-Client und X-Server

Durch die Trennung von X-Client und X-Server benötigt der Arbeitsplatzrechner weniger Ressourcen in Form von Arbeitsspeicher und Festplattenplatz. Er wird zu einem „dünnen“ Arbeitsplatzrechner, einem Thin Client. Das Thin Client Konzept macht von dieser Trennung Gebrauch. Auf dem Thin Client läuft der X-Server, der die Steuerung der Hardware bestehend aus Maus, Tastatur, Grafikkarte und Bildschirm übernimmt. Auf einem zentralen System, dem Terminalserver, laufen die X-Clients, deren Ausgaben durch das X Protokoll über das Netzwerk zum X-Server des Thin Clients weitergeleitet werden und vor dem Benutzer auf dem Bildschirm zur Darstellung kommen. Abhängig vom Durchsatzvermögen des Netzwerks ist es möglich das komplette Betriebssystem inklusive X-Server beim Start des Arbeitsplatzrechners über das Netzwerk zu laden. Das erspart den Festplattenspeicher. Zu Beginn eines Arbeitstages wird jedoch das Netzwerk übermäßig stark belastet, da beim fast zeitgleichen Hochfahren der Arbeitsplatzrechner durch die Mitarbeiter für jeden Rechner das Betriebssystem über das Netzwerk geladen werden muss. Bei einem schwachen Netzwerk ist es unumgänglich das Betriebssystem mit dem X-Server auf dem Thin Client zu speichern.

Durch Multicast TFTP kann die Last reduziert werden. Dies wird im Novell Produkt NLPOS bereits eingesetzt und zum Beispiel von Kaufhausketten für das morgendliche Inbetriebnehmen von mehreren hundert Kassensystemen über 10Mbit/s-Netze genutzt. Daneben kann komprimierte Übertragung das Netzwerk entlasten.

Für die physische Trennung von X-Server und X-Client ist ebenso ein leistungsfähiges Netzwerk für das X Protokoll notwendig, ansonsten wirkt die Darstellung auf dem Thin Client langsam und zäh.

Hierbei ist wiederum zu berücksichtigen, dass mehrere Mitarbeiter gleichzeitig an ihren Thin Clients arbeiten und in ständigem Kontakt mit dem zentralen System stehen.

3.2. Die NX Technologie

Die NX Technologie setzt an dem Bandbreitenhunger des X Protokolls an und verpasst ihm eine „Diät". *„NX ist ein[e] fast vollständig unter der GNU GPL stehende Remote-Desktop-Software der italienischen Firma NoMachine. Mit NX kann man den Bildschirminhalt eines entfernten Computers auf einen lokalen Rechner (auch betriebssystemübergreifend) übertragen und damit arbeiten."*[2] Abschnitt 3.1 beschreibt das X Window System als Client-Server-Modell, dessen Kommunikation über das X Protokoll abgewickelt wird. Für diese Kommunikation ist jedoch ausreichend Bandbreite notwendig und über Modem- oder ISDN-Verbindung erscheint dem Anwender die Anwendung eher zäh und langsam, was unter anderem durch die vielen Operations-Codes verursacht wird, die das X Protokoll definiert [BP04].

NX ist ein Aufsatz zum X Protokoll und steigert die Kommunikationseffizienz des X Protokolls durch folgende Maßnahmen:

- Die Übertragungsdaten werden komprimiert.
- Es wird ein Cache für bereits übertragene Daten angelegt.
- Die Roundtrips zwischen X-Client und X-Server werden reduziert.

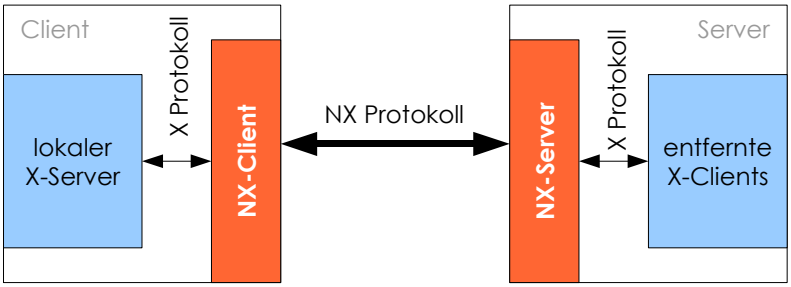

Abbildung 3.4.: Architektur der NX Technologie

Abbildung 3.4 zeigt die Architektur der NX Technologie. Eine entfernte X-Anwendung in Form eines X-Clients ist mit einem NX-Server über das X Protokoll auf dem Server verbunden (rechts im Bild). Die X-Anwendung kann auch eine vollständige Desktop Sitzung sein, wie beispielsweise KDE oder GNOME. Der NX-Server cached und komprimiert die Daten und schickt sie verpackt im NX-Protokoll an den Client (links im Bild). Der NX-Client entpackt und cached die Daten und gibt sie über das X Protokoll an den lokalen X-Server weiter.

Mit dem Client-Server-Modell von NX lassen sich ähnliche Szenarien konstruieren, wie mit dem X Window System. Abbildung 3.5 zeigt eine Beispielumgebung. Der NX-Server fungiert als „Proxy" für die Protokolle RDP und VNC und für das X Protokoll und kapselt sie in das NX Protokoll. Auf den Clients ist jeweils ein NX-Client installiert, der die Darstellung vom NX-Server auf den lokalen X-Server bringt. NX ermöglicht somit mit einem Client die Darstellung verschiedener Terminaldienste über dasselbe Protokoll. Hierbei ist es außerdem möglich das NX Protokoll verschlüsselt über SSH zu betreiben. Da in vielen Firewalls der SSH Port 22 freigeschaltet ist, entfällt das Öffnen der Firewall für einen weiteren Port.

[2] Artikel *NX*. In Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 23. Februar 10:00 UTC URL: `http://de.wikipedia.org/wiki/NX` (Abgerufen: 21. März 2006)

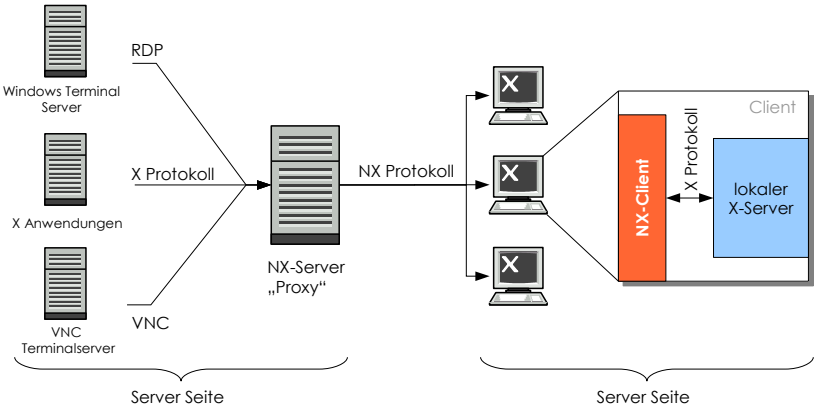

Abbildung 3.5.: NX Umgebung

Die NX Technologie vermindert den Netzwerkverkehr zwischen einem X-Client und einem X-Server und ermöglicht somit den physische Trennung dieser beiden über durchsatzschwache Leitungen, wie beispielsweise Modem, ISDN oder GPRS. Die Darstellung der Anwendung erscheint nicht mehr zäh, sondern läuft flüssiger ab, als nur über das X Protokoll. Der Anwender erlangt den Eindruck die Anwendung würde direkt auf dem Arbeitsplatzrechner vor laufen.

3.3. Terminalserver

Das zentrale System, auf dem die Anwendungen bei der physischen Trennung von X-Client und X-Server laufen wird im Bereich des Server-based-Computing als Terminalserver bezeichnet. Server-based-Computing umfasst die zentrale Verwaltung und Ausführung von Anwendungsprogrammen auf einem Server.

„Ein Terminalserver bezeichnet einen Computer, der mehrere Terminals [...] emuliert bzw. die Software, die besagte Emulation ermöglicht.“[3] Für eine Thin Client Umgebung ist ein Terminalserver als zentrales System für die Anwendungen notwendig. Auf ihm werden alle Anwendungen betrieben, die dem Thin Client zur Verfügung stehen.

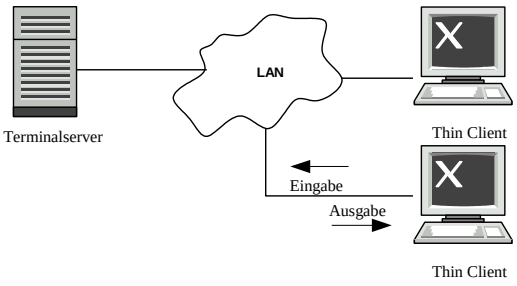

Abbildung 3.6.: Thin Client Umgebung

Für die Kommunikation zwischen Thin Client und Terminalserver existieren verschiedene Protokolle. Für Windows sind VNC und RDP die bekanntesten Terminalserverprotokolle. Unter Unix und ähnlichen Betriebssystemen kommt meist das X Protokoll des X Window Systems zum Einsatz.

Der Thin Client dient hierbei der Eingabe über Tastatur bzw. Maus und der Ausgabe über einen Bildschirm. Er leitet die entsprechenden Daten zum Terminalserver weiter und empfängt die Ausgabe zur Darstellung am Bildschirm. Abbildung 3.6 zeigt eine schematische Darstellung einer Thin Client Umgebung. Thin Client und Terminalserver sind über ein Local Area Network (LAN) miteinander verbunden. Die Benutzereingaben am Thin Client gehen über das Netzwerk zum Terminalserver, wo sie verarbeitet werden. Die Ausgabe geht über das Netzwerk zurück an den Thin Client, wo sie am Bildschirm angezeigt wird. Hierbei handelt es sich um ein *verteiltes* EVA-Prinzip (Eingabe

[3] Artikel *Terminalserver*. In Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 4. März 19:51 UTC URL: `http://de.wikipedia.org/wiki/Terminal_Server` (Abgerufen: 06. März 2006)

- Verarbeitung - Ausgabe). Ein- und Ausgabe erfolgen beim Thin Client, die Verarbeitung findet auf dem Terminalserver in einer konkreten Anwendung statt. Im Betriebssystem des Thin Clients findet ebenso ein Teil der Verarbeitung statt, hauptsächlich jene für die Umleitung von Ein- und Ausgabe.

Terminalserver gibt es für Windows, Unix und GNU/Linux. Der X-Server des Thin Clients verbindet sich hierbei beispielsweise mit dem grafischen Anmeldebildschirm des Display Managers auf dem Terminalserver und stellt ihn dem Anwender des Thin Clients zur Verfügung, so dass er sich einloggen und einen Desktop benutzen kann.

3.4. Server-Backend

Als Server-Backend für den Terminalserver in Schwäbisch Hall dient eine IBM OpenPower, mit der die verschiedenen Linux-Distributionen virtualisiert werden. Abbildung 3.7 zeigt eine schematische Darstellung der Virtualisierung. Der Hypervisor sitzt zwischen der physischen Hardware und den zu virtualisierenden Betriebssystemen. Er wacht über die Verteilung der Ressourcen (CPU-Zeit, I/O-Zyklen) der Gastsysteme. Der Hypervisor wird durch Linux verwaltet (linke Seite in der Abbildung). Die Virtualisierung erlaubt den parallelen Betrieb mehrerer Betriebssysteme in isolierten Umgebungen, so als ob jedes System auf eigener physischer Hardware installiert ist. Durch Virtualisierung wird die Hardware effizienter genutzt.

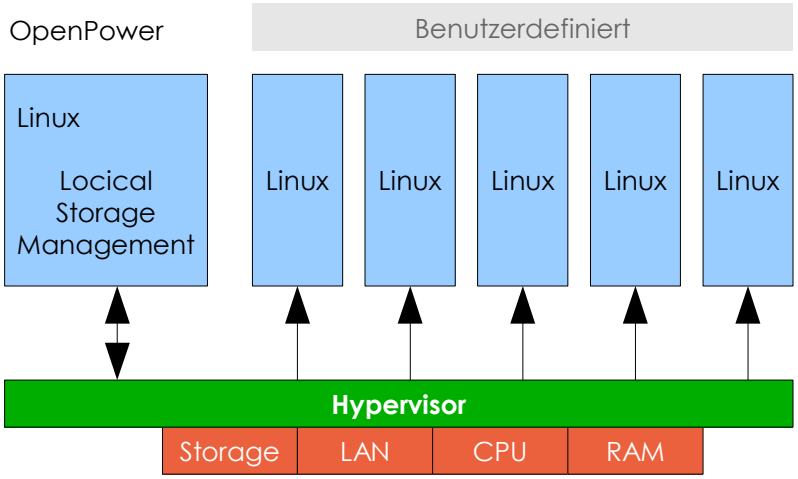

Abbildung 3.7.: Virtualisierung auf IBM OpenPower

Die verschiedenen Linux-Installationen stellen unterschiedliche Desktops zur Verfügung, die zentral von einem NX-Server angesprochen und den Thin Clients als Terminaldienst bereit gestellt werden.

Somit ergibt sich für die Stadt Schwäbisch Hall die schematische Server-Backend Struktur wie in Abbildung 3.8 dargestellt. Die OpenPower Maschine virtualisiert verschiedene Linux Desktops, zum Beispiel GNOME und KDE. Des Weiteren läuft in einer Virtualisierung der NX-Server, der als Proxy für die Terminalserver dient. Die Thin Clients verbinden sich über den NX-Client mit dem NX-Server. Entsprechend der Auswahl im NX-Client verbindet sich der NX-Server zu einem der Linux Desktops oder zu einem der Windows-Versionen auf dem Intel-Server.

3.5. Thin Client

Nach der Beschreibung eines allgemeinen Arbeitsplatzrechners, der physischen Trennung X-Client und X-Server, der Optimierung des X Protokolls durch die NX Technologie, der Definition und Be-

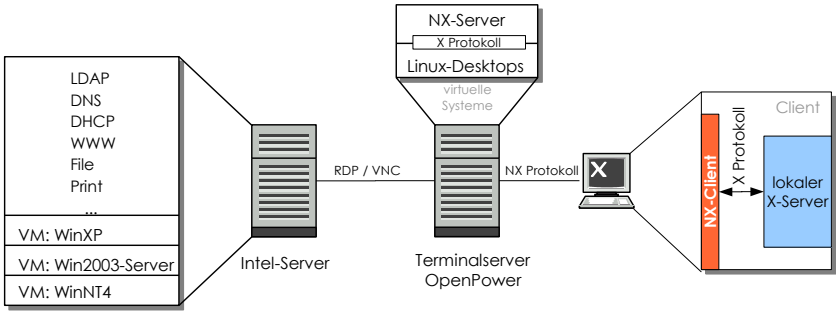

Abbildung 3.8.: Struktur Server Backend

schreibung des Terminalservers in Schwäbisch Hall sei nun der Kern dieses Dokuments beschreiben, der Prototyp des Open Source basierten Thin Clients der Stadt Schwäbisch Hall.

„Das Thin-Client-Konzept bedeutet, dass ein Client seine Daten möglichst vollständig von einem Server bezieht“[4]. Ein Thin Client im Kontext dieses Dokuments besteht aus Hardware, die über einen Prozessor (CPU), Arbeitsspeicher (RAM), eine Netzwerkkarte (LAN), eine Grafikkarte (Display) und nicht-flüchtigen Speicher (Storage), wie in Abbildung 3.9 dargestellt, verfügt. Der nicht-flüchtige Speicher kann über verschiedene Anschlüsse, wie USB, IDE oder SD zur Verfügung gestellt werden.

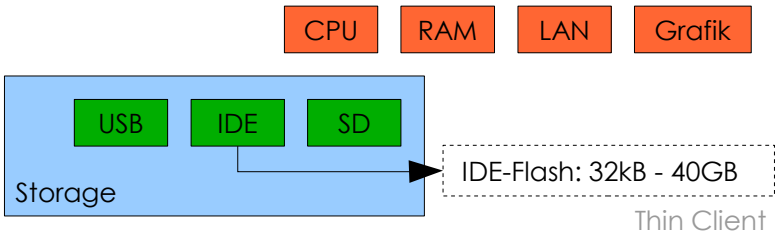

Abbildung 3.9.: Schematische Darstellung der Architektur eines Thin Clients

Von der Softwareseite aus betrachtet, befindet sich auf dem Thin Client ein Betriebssystem und ein X-Server. Der Thin Client entspricht dem Arbeitsplatzrechner, auf den zu Beginn des Dokuments eingegangen wurde. Alle weiteren Anwendungen bezieht der Thin Client über das Netzwerk von einem zentralen System. Anwendungen werden zentral auf einem oder mehreren Servern ausgeführt und nur die Bildschirmausgabe und die Tastatur- bzw. Mauseingabe werden zwischen Server und Client über das Netzwerk mit einem Protokoll übertragen.

Folgende Vorteile ergeben sich beim Einsatz von Thin Clients:

- Kleine Desktop Computer
- Können über das Netzwerk booten
- Benötigen keine Festplatten, sogenannte Diskless Thin Clients
- Niedriger Stromverbrauch
- Geringe Wärmeentwicklung
- Geringere Kosten als ein traditioneller Desktop-PC
- Niedrige bis keine Geräuschentwicklung (abhängig davon, ob bewegliche

[4] Artikel *Thin Client*. In Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 18. Februar 18:51 UTC URL: `http://de.wikipedia.org/wiki/Thin_client` (Abgerufen: 02. März 2006)

- Teile in
- Form von Lüftern eingebaut sind)
- Zentrale Administration
- Verwenden Standardhardware
- Entfernter Zugriff auf Desktop
- Recycling alter PCs zu Thin Clients

Listen zu Hard- und Softwareherstellern für Thin Clients befinden sich in den Anhängen A.1 und A.2. Thin Clients ohne Storage-Komponente werden *diskless Thin Clients* genannt. Sie verfügen über die Komponenten CPU, RAM, LAN und Display und booten ihr Betriebssystem über das Netzwerk vom Terminalserver. Der Vorteil besteht darin, dass für eine Reihe von Thin Clients nur eine Betriebssystem-Image auf dem Terminalserver bereitgehalten und gepflegt werden muss. Allerdings ist ein Netzwerk mit entsprechender Bandbreite unbedingt notwendig, da bei jedem Hochfahren eines Thin Clients das Betriebssystem-Image inklusive X-Server über das Netzwerk übertragen und in den RAM des Thin Clients geladen werden muss.

Es gibt eine Vielzahl von Hardwareherstellern für Thin Clients, die jedoch keinen installierten NX-Client im Angebot haben und somit das NX Protokoll nicht unterstützen. Die linuxbasierten Thin Clients verfügen über proprietäre Erweiterungen. Jeder Hersteller verwendet eigene Lösungen für deren Verwaltung. Es ist kein Management für eine heterogene Thin Client Umgebung bekannt. Gemischte Thin Client Umgebungen lassen sich somit nicht mit einer Software zentral administrieren. Des Weiteren sind die Clients stark herstellerspezifisch und verwenden keine einheitlichen Softwareimages. Die Linux-Systeme verfügen meist noch über einen veralteten Kernel der 2.4er Reihe.

Neben Hardwareherstellern für Thin Clients gibt es verschiedene Projekte, die sich dem Thema Server-based-Computing auf Softwareseite widmen, von denen die folgenden drei kurz betrachtet wurden:

- Linux Terminal Server Project (LTSP)
- PXES
- Thin Station

Alle drei Projekte bieten Thin Client Funktionalität basierend auf Linux. Tabelle 1 gibt einen Überblick über die benutzten Kernelversionen und die Boot-Methoden der untersuchten Thin Client Projekte. Sie legen den Fokus speziell auf Diskless Thin Clients, die ausschließlich über das Netzwerk gebootet werden. Nach Abschluss des Bootvorgangs verbindet sich der X-Server des Thin Clients mit dem Terminalserver, der den Desktop zur Verfügung stellt.

	LTSP	PXES	Thin Station
Version	4.1.1	1.0	2.0
Kernelversion	2.6er	2.4er	2.4er
Boot-Methoden	Etherboot, PXE, RPL	PXE	Etherboot, PXE
Nötige Netzwerkdienste	DHCP, TFTP	DHCP, TFTP	DHCP, TFTP

Tabelle 3.1.: Überblick Thin Client Projekte

Das LiSoG-Szenario verlangt die Verbindung mit einem NX-Server, der neben dem X Protokoll auch die Protokolle RDP und VNC unterstützt. Nur das PXES Projekt unterstützt das NX-Protokoll.

Der Prototyp des Open Source basierten Thin Clients verfügt über eine Storage-Komponente, auf der Betriebssystem und X-Server installiert sind. Somit entfällt das Laden eines Betriebssystem-Images über das Netzwerk beim Booten dieses Thin Clients. Die Anwendungen befinden sich zentral auf einem Terminalserver, mit dem sich der Thin Client verbindet.

In Schwäbisch Hall kommen als Storage-Komponenten derzeit IDE-Flash Speicher zum Einsatz. Sie sind in Größen zwischen 32 kB und 40 GB erhältlich. Der Prototyp verfügt über folgende Hardwareausstattung:

CPU: VIA C3 bzw. Transmeta, Taktraten zwischen 400 – 800 MHz

RAM: 128 – 384 MB

HD: 1 – 5 GB

Grafik: 1024x768, 75 Hz, 32 Bit

3.6. Thin Client Software

Die Basissoftware des Thin Clients besteht komplett aus Freier Software und Open Source Komponenten. Sie umfasst einen Linuxkernel mit entsprechenden Modulen für die Storage-Komponenten (USB-Unterstützung, IDE-Unterstützung, SD-Unterstützung) und einen X-Server. Der Prototyp verwendet Ubuntu 5.04 bzw. 5.10 in der vom Installer angebotenen Serverinstallation. Diese Paketierung installiert die wenigsten Pakete. Sie wurde weiter reduziert, so dass das System auf dem Thin Client nur noch aus wirklich notwendigen Komponenten für den Betrieb der Hardware und den Einsatz des X-Servers installiert sind. Die Installation benötigt ca. 300 MB Speicherplatz auf der Storage-Komponente und kann weiter reduziert werden. Wenn zusätzlich die deutschen Sprachpakete, der Window Manager IceWM, ein Browser und ein NX-Client installiert werden, benötigt die Installation ca. 450 MB.

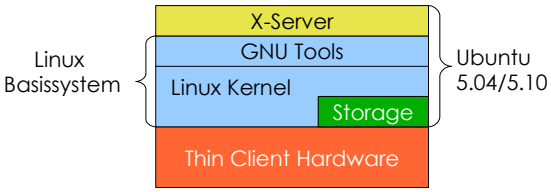

Abbildung 3.10.: Softwarebasis Thin Client

Abbildung 3.10 zeigt den Software-Stack für den Thin Client. Für die Ausführung von grafischen Programmen ist lediglich ein X-Server notwendig. Ein gestarteter xterm im reinen X-Server hat keinen Fensterrahmen oder Symbole zum Maximieren oder Minimieren. Diese Funktionalitäten zum Verändern der Fenstereigenschaften sind im Window Manager angesiedelt. Ein traditioneller linuxbasierter Thin Client verfügt prinzipiell über denselben Software-Stack. Auch die Open Source Projekte zum Thema Server-based-Computing implementieren den Ansatz aus Abbildung 3.10. Eine XDMCP-Anfrage an den Display Manager des Terminalservers erzeugt einen grafischen Anmeldebildschirm auf dem Bildschirm des Thin Clients. Der Display Manager und der dazugehörige Window Manager laufen auf dem Terminalserver.

Mit der Softwarebasis aus Abbildung 3.10 ergeben sich verschiedene Möglichkeiten den Thin Client um Komponenten in Abbildung 3.11 zu erweitern. Der betrachtete Mininalansatz in Abbildung 3.11 ist mit 1. markiert. Er sieht zusätzlich zur beschriebenen Open Source Softwarebasis einen NX-Client vor. Der NX-Client ermöglicht die Verbindung zu einem NX-Server, der mit seiner Proxytätigkeit auf verschiedene Terminalserver mit den Protokollen X, RDP und VNC zugreift. Über den NX-Server können dem Benutzer am Thin Client verschiedenste Desktop-Systeme zur Verfügung gestellt werden.

Möglichkeit 2 in Abbildung 3.11 sieht weitere Softwarekomponenten für den Thin Client vor. Mit Internetzugang, einem Window Manager und einem Webbrowser wird aus dem Thin Client ein einfacher Internetclient, der ohne Zugriff auf einen Terminalserver auskommt und somit auch *offline* arbeiten

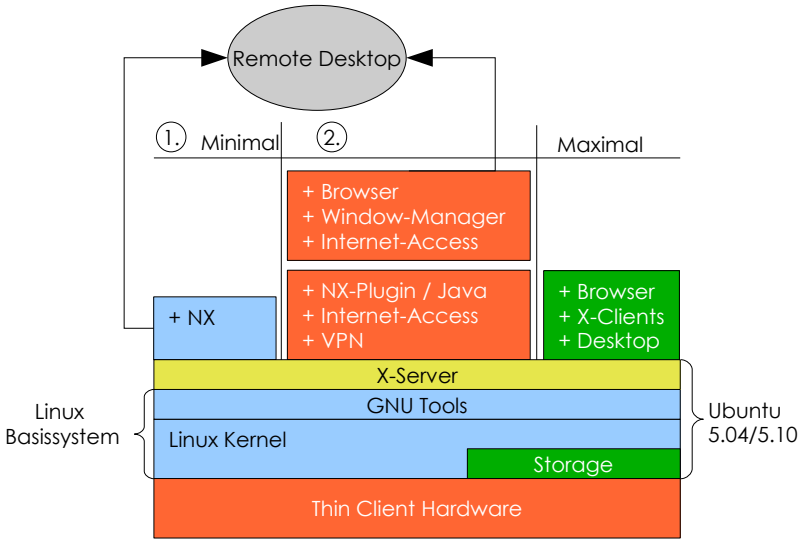

Abbildung 3.11.: Thin Client Ansätze

kann. Mit zusätzlichem NX-Plugin für den Browser und VPN ist auch ein Zugriff auf den NX-Server möglich, über den, wie in Möglichkeit 1 ein Remote Desktop bereit steht.

Der Maximalansatz hat zusätzlich zu der Softwarebasis ein komplettes Desktopsystem mit Browser und X-Clients. Die Desktop-Anwendungen werden nicht mehr auf dem Terminalserver gestartet und ausgeführt, sondern auf dem Thin Client selbst. Die Ausstattung moderner Thin Client Hardware erlaubt den lokalen Betrieb eines Desktop-Systems, setzt allerdings genügend Speicherkapazität auf der Storage-Komponente voraus. Dieser beschriebene Ansatz verlässt die Grunddefinition für den Thin Client und macht aus ihm einen Fat Client, der neben Ein- und Ausgabe auch die Verarbeitung übernimmt. Abbildung 12 zeigt hierzu eine Einordnung verschiedener Clientsysteme. Möglichkeit 1 aus Abbildung 11 passt lediglich in die Einordnung für einen Thin Client. Möglichkeit 2 und der Maximalansatz sind schon eher im Bereich Fat Client anzusiedeln, da sie mit einen Browser bereits eine zusätzliche Verarbeitung auf der Hardware über das Betriebssystem hinaus durchführen.

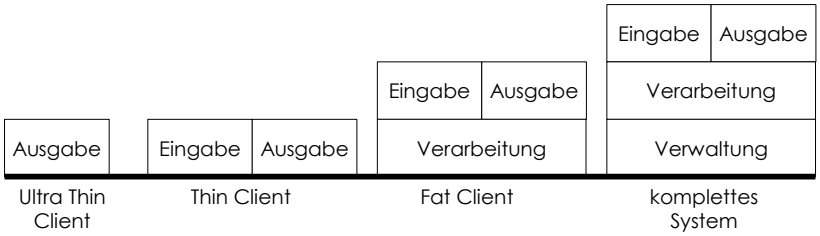

Abbildung 3.12.: Einordung von Clientsystemen (Quelle: Artikel Thin Client. In Wikipedia, Die freie Enzyklopädie.)

Die in Abbildung 3.11 aufgezeigte Möglichkeit 1 wurde auf einem 1 GB IDE-Flash realisiert. Auf 2 GB kann ein kompletter Desktop installiert werden, zum Beispiel Ubuntu oder Xandros. Als Storage-Komponente wurden auch 2 bzw. 4 GB USB-Flashs ausprobiert, auf denen ebenfalls Ubuntu installiert wurde. Stehen 5 GB zur Verfügung, so kann sogar eine SuSE 9.3 auf der Hardware des Thin Clients installiert werden. Doch dadurch wird aus dem Thin Client ein komplettes System, wie in Abbildung 3.12 dargestellt.

Der Prototyp wurde mit Ubuntu 5.04 / 5.10 in der Paketauswahl *Server* realisiert, der um die Pakete für einen X-Server erweitert und gemäß der Möglichkeit 1 aus Abbildung 11 angepasst wurde. Somit wurde ein komplett linuxbasierter Open Source Thin Client bestehend aus Standard Software-

Komponenten beschrieben. Durch die Verwendung von Ubuntu steht ein System zur Verfügung, dass ausschließlich Open Source und Freie Software verwendet, einen aktuellen Kernel der Version 2.6.x zur Verfügung stellt, leicht paketierbar und für die Massenproduktion geeignet ist.

4. Anzugehende Problemlösungen

Mit dem Prototyp ist die Grundlage für darauf aufbauende Arbeiten geschaffen. Dieser Abschnitt möchte Defizite aufzeigen, die es aus Sicht der LiSoG zu bearbeiten gilt.

4.1. Benchmarking

Das Szenario beschreibt den Aufbau eines Open Source basierten Thin Clients und seine architektonische Integration in eine entsprechende Infrastruktur. Ein besonderes Augenmerk liegt auf dem NX Protokoll und der damit verbundenen Technologie mit ihrem Vorteil der Bandbreiten schonenden Übertragung, so dass Verbindungen zum NX-Server über GPRS, Modem und ISDN möglich sind. Getestet wurde das Verhalten mit einem Prototyp.

In produktiven Umgebungen sind jedoch selten wenige Thin Clients parallel aktiv. Für die Stadt Schwäbisch Hall stellt sich deswegen die Frage, wieviele NX-Sessions parallel über die OpenPower betrieben werden können und regt ein Benchmark an. Interessant wäre ein Vergleich mit einem Intel-Server. Hierfür gilt es einen Benchmarking-Experten zu finden, der sich bereit erklärt, das Benchmark zu konzeptionieren und durchzuführen.

Novell konnte in Berlin die Leistung von Linux-Terminalservern (SLES9 x86_64, KDE, OpenOffice, Mail, Browser) im produktiven Betrieb mit mehreren hundert Nutzern beobachten. Basierend darauf können folgende Regeln für das überschlägige Sizing von Linux Terminalserver genutzt werden:

Pro 50 concurrent user:

- 1 CPU 2.4-3 GHz (AMD Opteron 2.4 GHz bzw. Intel P4 EM64T 3 GHz)
- 4 GB RAM
- 2-4 GB Swap Space
- 1-2 GB /tmp (Abhängig von der Anzahl der Benutzer)

4.2. Hard- und Software

4.2.1. Hersteller für Standard Thin Client gesucht

Der Prototyp besteht aus Standard Open Source Software-Komponenten. Die Paketierung wird bereits von der verwendeten Linux Distribution übernommen. Es fehlen Hersteller, die die beschriebenen Standard-Komponenten als ein Software Image auf entsprechende Flash-Speicher spielen und anbieten bzw. Hersteller für Thin Clients, die die Hardware bereits mit der Software-Ausstattung liefern.

Ein Thin Client Hersteller kann diesen linuxbasierten Open Source Thin Client zusätzlich in sein Produktportfolio aufnehmen und so Kunden bedienen, die explizit vollständig Open Source für den Thin Client fordern. Solche Kunden können beispielsweise Kommunen wie Schwäbisch Hall und öffentliche Einrichtungen sein. Die Paketierung der Standard Open Source Software wird bereits durch die Distribution geleistet. Das benötigte Software Image kann vom Hersteller selbst erstellt und auf die

Hardware aufgespielt werden. Der Mehraufwand in der Produktlinie hält sich hierfür in Grenzen. Das Geschäftsmodell könnte so aussehen, dass der Hersteller Support und Wartung der Geräte anbietet. Der Support der Software wird durch die verwendete Distribution abgedeckt, so dass sich der Support des Herstellers auf die Hardware beschränken lässt.

4.2.2. Systems Management für Thin Clients gesucht

Thin Client Hersteller bieten für ihre Produkte eine Reihe von eigenen Erweiterungen zur zentralen Verwaltung der Thin Clients an. Diese Erweiterungen sind meist proprietär und erlauben nur das Management der Hersteller-eigenen Produkte und schließen heterogene Umgebungen aus. Beim Kauf von Thin Clients möchte sich ein Anwender selten auf einen Anbieter festlegen und seine Infrastruktur mit Thin Clients von nur einem Hersteller aufbauen lassen. Unter Umständen verfügt der Anwender bereits über Thin Clients bzw. funktioniert alte PCs zu Thin Clients um. Damit entsteht eine heterogene Thin Client Umgebung, die zentral verwaltet werden soll. Für dieses Szenario reicht die Management-Software eines Thin Client Herstellers nicht mehr aus, um diesen „Zoo“ zu verwalten.

Derzeit ist den Autoren keine Open Source Software für die zentrale Verwaltung von Thin Clients, deren Benutzern, Softwareverteilung und Patch Management bekannt. Es ist allerdings möglich aus existierenden Bausteinen wie LDAP, Open Source Datenbanken, Scripten und anderen Werkzeugen ein solches System zu bauen. Benötigt wird eine Lösung, die o.g. Punkte abdeckt. Zu klären sind Details und Anforderungen für ein Systems Management für Thin Clients. Die LiSoG bearbeitet seit März 2006 das Szenario Desktop Management, aus dem sich einige Punkte des Anforderungskatalogs verwenden lassen, so zum Beispiel Single Point of Administration, zentrales Benutzer-, Gruppen- und Thin Client bzw. Computer-Management.

Novell bietet beispielsweise mit seinem Produkt NLPOS[1] ein Management-Framework für Thin Clients, das großteils aus Open Source Komponenten aufgebaut ist.

Auch Univention bieten mit seinem Univention Corporate Server[2] eine Linux-Komplettlösung, mit der Standort- und Plattformübergreifend nicht nur Linux-basierte Systeme, sondern auch heterogene Infrastrukturen verwaltet werden können. Zentrale Komponenten des Managementsystems sind ein LDAP-Verzeichnisdienst und Kerberos-Dienste, so dass sich vorhandene Dienste und Anwendungen sehr einfach integrieren lassen.

Eine genaue Untersuchung der genannten Vertreter bzw. anderer Lösungen steht noch aus.

4.3. Technische Implementierungen

4.3.1. USB over IP

Das Thema USB over IP adressiert die Problematik lokal an einen Thin Client angeschlossene USB-Geräte vom Terminalserver aus, wo die Anwendung zur Handhabung von USB läuft, anzusteuern. Beispielsweise soll es möglich sein einen Drucker oder einen Scanner lokal an den USB-Port des Thin Clients anzuschließen und über entsprechende Anwendungen des vom Terminalserver bereitgestellten Desktops zu steuern.

Problem hierbei ist die Trennung des USB-Systems von der verarbeitenden Anwendung. Abbildung 4.1 zeigt die Problemstellung. Soll beispielsweise eine Anwendung auf die Daten einer Digitalkamera, die über USB angeschlossen ist, zugreifen, so erfolgt der Zugriff über das Betriebssystem direkt auf die Hardware. In Abbildung 4.1 ist die Kamera jedoch an die Hardware des Thin Clients angeschlossen und die dazugehörige Anwendung läuft entfernt auf einem Terminalserver. Diese Anwendung kann

[1] NLPOS - Novell Linux Point of Service: `http://www.novell.com/documentation/nlpos9/`
[2] Univention Corporate Server: `http://www.univention.de/ucs.html`

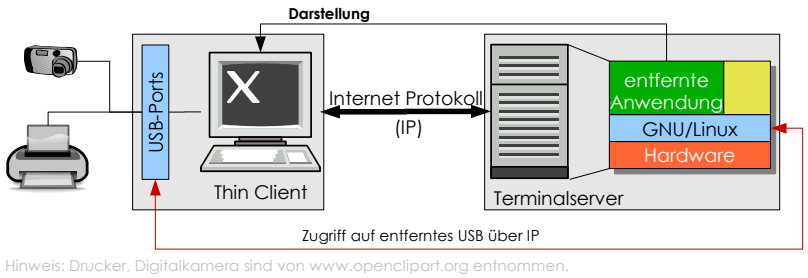

Abbildung 4.1.: USB über Internet Protokoll (USB over IP)

nicht über das Betriebssystem direkt auf die Hardware zugreifen, weil die Digitalkamera nicht an den Terminalserver angeschlossen ist. Die angeschlossene Digitalkamera muss der entfernten Anwendung als lokal angeschlossenes Gerät bekannt gemacht werden, damit der Anwender am Thin Client damit arbeiten kann.

Das USB/IP Projekt befasst sich mit dieser Problemstellung[3]. Das Projekt verfügt über einen Entwickler aus Japan und bezeichnet sich selbst als Pre-Alpha[4]. Eine Unterstützung, insbesondere in personeller Hinsicht, könnte das Projekt weiter voranbringen und beschleunigen. Die Funktionalität konnte bisher allerdings noch nicht auf ihre Benutzbarkeit getestet werden. Eine Untersuchung auf Benutzbarkeit steht noch aus.

4.3.2. USB/SIM Authentisierung

Mit der USB/SIM Authentisierung soll eine Smartcard-ähnliche Authentisierung vorgenommen werden. Ein USB-Stick besteht hierbei aus den beiden Komponenten Flash-Speicher und eines SIM-ähnlichen Chips. *„Die SIM-Karte (Subscriber Identity Module) ist eine Chipkarte, die in ein Mobiltelefon eingesteckt wird und zur Identifikation des Nutzers im Netz dient.“*[5] Analog zum Konzept der SIM-Karte für Mobiltelefone sind auf dem Chip im USB-Token eine digitale Signatur und ein Zertifikat gespeichert, die den Benutzer des USB-Tokens eindeutig identifizieren. Der USB-Token soll die Funktionalität eines Security-Token erfüllen. *„Ein Security-Token bezeichnet eine Hardwarekomponente, die in der Regel eine Chipkarte enthält, aus der keine Daten herauskopiert oder manipuliert werden können.“*[6] Die Daten auf dem USB-Token sind notwendig zur Authentisierung des Benutzers gegenüber dem System, im vorliegenden Fall gegenüber eines Thin Clients. Mit dem Zertifikat auf dem Chip sind zusätzlich die Daten auf dem Flash-Speicher verschlüsselt und der Zugriff darauf wird durch den Chip geregelt. Über eine PIN, ein Passwort oder über biometrische Informationen wird der Zugriff auf die Daten im Flash-Speicher freigegeben.

Vereinfacht gesagt handelt es sich hierbei um eine Kombination aus Smartcard und USB-Stick. Die Smartcard dient der Authentisierung und der USB-Stick der Speicherung von Daten. Die Kombination erlaubt die Verschlüsselung der Daten auf dem USB-Stick, die mit zusätzlichen Informationen, wie PIN, Passwort oder biometrische Informationen entschlüsselt werden.

Die Idee besteht darin, die Authentisierung am Thin Client über ein solches USB-Token zu realisieren. Es gibt Bausteine mit deren Hilfe eine Lösung zusammengestellt werden kann. Es fehlt jedoch an einem integrierten Produkt. Die Anregung besteht darin, eine solche integrierte Lösung für die Authentisierung am Thin Client in einer linuxbasierten Open Source Thin Client Umgebung umzusetzen,

[3] `http://usbip.naist.jp/`
[4] `http://sourceforge.net/projects/usbip`
[5] Artikel *SIM-Karte*. In Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 15. Februar 8:15 UTC URL: `http://de.wikipedia.org/wiki/SIM-Karte` (Abgerufen: 31. März 2006)
[6] Artikel *Security-Token*. In Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 16. Februar 19:49 UTC URL: `http://de.wikipedia.org/wiki/Security-Token` (Abgerufen: 31. März 2006)

die wiederum Open Source ist. Die Firma charismathics[7] vertreibt USB-Sticks mit einer kombinierten Smartcard Funktionalität.

[7] http://www.charismathics.com/products/1for_business_customers/3plug--in--icrypt.php

5. Fazit

Der Thin Client Prototyp der Stadt Schwäbisch Hall zeigt, dass die Bestandteile für einen linuxbasierten Open Source Thin Client vorhanden sind. Linux und das X Window System bieten hervorragende technische Voraussetzungen für eine Thin Client Umgebung, die insbesondere bei Standardarbeitsplätzen wie beispielsweise Office mit E-Mail und Internet Sinn macht, da hier keine spezielle Software benötigt wird und die Hardwareanforderungen nicht besonders hoch sind. Der Arbeitsplatz kann zentral verwaltet werden und verringert somit den Administrationsaufwand.

Wirtschaftlich ermöglicht eine Thin Client Umgebung die weitere Nutzung ausgedienter PCs, indem sie zu Thin Clients umfunktioniert werden. Somit können PCs über fünf Jahre oder länger anstatt der üblichen drei Jahre genutzt werden. Darüber hinaus ist die Thin Client Hardware selbst preisgünstiger und wartungsfreier als PC Hardware.

Durch die NX Technologie erfährt der Anwender keine Einschränkungen in der Performance der Anwendungen gegenüber einem PC. Der Schritt zum ortsungebundenen Arbeiten ist mit NX nicht mehr weit. Wird anstatt eines ausgedienten PCs normale Thin Client Hardware eingesetzt reduziert sich zusätzlich der Geräuschpegel der Hardware.

Der Prototyp der Stadt Schwäbisch Hall zeigt jedoch Defizite bei linuxbasierten Thin Clients auf. So fehlt es beispielsweise an einem Hersteller oder IT Dienstleister, der Thin Clients mit den Software Images ausliefert und auch Wartung und Support für die Geräte anbietet. Der Software Support kommt hierbei von der Distribution, im beschriebenen Fall kostenlos von Ubuntu.

Des Weiteren fehlt es an einer zentralen Systems Management Software Lösung für Thin Clients, die verschiedene Thin Clients auch über Distributionsgrenzen hinweg verwaltet. Auch technisch bietet ein Open Source Thin Client weitere Herausforderungen, wie beispielsweise USB over IP oder USB/SIM Authentisierung.

Literaturverzeichnis

[BP04] Horst Bräuner and Kurt Pfeifle. *NX - Turbolader für Linux- und Windows-Terminalserver.* `http://www.pl-berichte.de/berichte/lt2004-nxartikel.html`, 2004.

[Wik06] Wikimedia Foundation Inc., `http://de.wikipedia.org/`. *Wikipedia – Die freie Enzyklopädie*, 2006.

A. Hersteller

A.1. Thin Client Hardware

- A Thin Client Company: www.athinclient.net
- ATLabs: www.atlabs.com
- Affirmative: www.affirmative.de
- BB-5000: www.bb-5000.info/data/products/tc/
- BOScom Ltd.: www.bosweb.com/thinclients.htm
- Chip PC: www.chippc.com
- Computer Lab International: www.computerlab.com
- DT Research: www.dtresearch.com
- EIZO: eclient.eizo.com
- Esprit Systems: www.espritsys.com/productthin.html
- Fujitsu Siemens: www.fujitsu-siemens.com
- HCL Peripherals: www.hclperipherals.com
- Hewlett Packard: h18006.www1.hp.com/products/thinclients/
- Igel Technology: www.igel.de
- JAS: www.jas-solution.com
- Levigo: www.levigo.de/thinclients
- Neoware Systems: www.neoware.com/thin-clients/
- Netvoyager: www.netvoyager.co.uk
- Nexterminal: www.nexterminal.com
- NLynx Technologies, Inc.: www.nlynx.com
- Rangee: www.rangee.de
- SmartFlex: www.smartflextech.com
- Sun: www.sun.de
- Thinner: www.thinner.de
- VECMAR Computer Solutions: www.vecmar.com/thin_client/thin_client.htm
- VXL Instruments: www.vxl.net/clients/clients.html
- WETIF: www.wetif.com
- Wyse Technology: www.wyse.com/thincomputing/

A.2. Thin Client Software

- Citrix Metaframe: www.citrix.com
- Microsoft Terminal Server: www.microsoft.com
- Tarantella Secure Global Desktop: www.tarantella.com
- GraphOn GO-Global: www.graphon.com
- Jetro CockpIT: www.jp-inc.com
- Softricity SoftGrid Universal Desktop Client: www.softricity.com
- Sun Ray: www.sun.com
- NTA NTAVO: www.ntavo.com
- HOB HobLink JWT (Java): www.hobsoft.com
- ThinSoft Inc. WinConnect: www.thinsoftinc.com
- Dat Group Panther Server: www.pantherpowered.com
- Menta Software WinToNet (Java): www.mentasoftware.com
- NoMachine: www.nomachine.com
- 2X Terminal Server: www.2x.com
- ShaoLin Aptus: www.shaolinmicro.com
- Linux Terminal Server Project: www.ltsp.org
- LTSP Supplementals (Webmin): termserv.berlios.de
- Symbiont Management Suite: www.thesymbiont.com
- SafeDesk: www.safedesksolutions.com
- PXES: pxes.sourceforge.net
- Thinstation: thinstation.sourceforge.net
- Ndiyo: www.ndyio.org
- OPTion Virtual Client for Linux: www.smartflextech.com
- Ericom PowerTerm: www.ericom.com
- RDesktop: www.rdesktop.org
- tsclient: www.gnomepro.com
- Gonicus GOto: oss.gonicus.de
- Dass-IT Smart Client Framework: www.dass-it.de
- NLPOS - Novell Linux Point of Service www.novell.com/documentation/nlpos9

A.3. Berichte

- Abschlußbericht Open4Future: http://www.parlament-berlin.de/ados/Haupt/vorgang/h15-3130.C.Anlage-v.pdf