Ein Referenzmodell für Server Based Computing Infrastrukturen und dessen Anwendung für das Kapazitätsmanagement

Dissertation

zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften

(Dr. rer. nat.)

durch die Fakultät für Wirtschaftswissenschaften der

Universität Duisburg-Essen

Campus Essen

vorgelegt von

Dipl.-Inform. (FH) Christian Knermann

aus Alpen, geboren in Kamp-Lintfort

Essen, 2016

Tag der mündlichen Prüfung: 22.06.2016

Erstgutachter: Prof. Dr. Bruno Müller-Clostermann

Zweitgutachter: Prof. Dr. Klaus Echtle

„There will always be plenty of things to compute in the detailed affairs of millions of people doing complicated things. (…) Every time one combines and records facts in accordance with established logical processes, the creative aspect of thinking is concerned only with the selection of the data and the process to be employed, and the manipulation thereafter is repetitive in nature and hence a fit matter to be relegated to the machines. (…) Whenever logical processes of thought are employed – that is, whenever thought for a time runs along an accepted groove – there is an opportunity for the machine."

– aus „As we may think" von Dr. Vannevar Bush
in The Atlantic Monthly, Band 176, Juli 1945

Vorwort

Die vorliegende Arbeit entstand während meiner Tätigkeit als Leiter der Gruppe „IT-Services“ am Fraunhofer-Institut für Umwelt-, Sicherheits- und Energietechnik UMSICHT in Oberhausen. Sie wurde angeregt durch zahlreiche Forschungs- und Industrieprojekte, in denen Terminal Server, virtuelle Desktops und Thin Clients selbst Forschungsgegenstand waren oder als unterstützende Technologien zum Einsatz kamen.

Für den Erfolg einer solchen Arbeit bedarf es der Unterstützung vieler Personen, denen ich an dieser Stelle danken möchte.

Mein besonderer Dank gilt Herrn Prof. Dr. Bruno Müller-Clostermann, dem ehemaligen Leiter des Fachgebiets „Systemmodellierung“ am Institut für Informatik und Wirtschaftsinformatik der Universität Duisburg-Essen, sowohl für seine wohlwollende Unterstützung als auch die stete Förderung meiner Arbeit und die Übernahme des Erstgutachtens.

Ebenso danke ich Herrn Prof. Dr. Klaus Echtle, dem Leiter des Fachgebiets „Verlässlichkeit von Rechensystemen“ am Institut für Informatik und Wirtschaftsinformatik der Universität Duisburg-Essen, für die Unterstützung als Zweitgutachter und für seine wertvollen Hinweise zu einer Vorversion dieser Arbeit.

Insbesondere bei Thorsten Wack und Andreas Schröder möchte ich mich für Diskussionen und konstruktive Anmerkungen zu Vorversionen bedanken sowie generell für die Schaffung eines Arbeitsumfelds, das dieses Vorhaben erst ermöglicht hat. Verena Buhle und Alexandra Emmerich übernahmen die Durchsicht und die Fehlerkorrektur des Manuskripts, wofür ich mich ebenso herzlich bedanke.

Darüber hinaus möchte ich allen Kolleginnen und Kollegen am Fraunhofer UMSICHT meinen Dank aussprechen, die über die Zusammenarbeit in der täglichen Arbeit direkt oder indirekt zum Gelingen dieser Arbeit beigetragen haben.

Mein Dank gilt auch meinen Eltern Norbert und Irmgard Knermann, die mich immer unterstützt haben und somit den Grundstein für meinen beruflichen und wissenschaftlichen Werdegang gelegt haben.

Vor allem möchte ich mich bei Janine Lehan bedanken, die mich während des gesamten Promotionsverfahrens unterstützt und motiviert hat. Ihre große Geduld und ihr Verständnis auch an arbeitsintensiven Wochenenden hat die Erstellung dieser Arbeit möglich gemacht.

Alpen, im Juni 2016 Christian Knermann

Kurzfassung

Der weltweit rasant steigende Bedarf an Unterstützung von Anwendern durch leistungsfähige IT-Systeme führt zu einer gleichermaßen steigenden Nachfrage nach Technologien, die es Unternehmen ermöglichen, ihren Endanwendern Desktop-Umgebungen und Applikationen in effizienter und effektiver Weise bereitzustellen. Daraus leitet sich sowohl unter ökologischen als auch unter ökonomischen Aspekten die Anforderung ab, vorhandene Hardware- und Software-Plattformen möglichst passend zum heutigen und zukünftigen Bedarf zu dimensionieren und die Systeme optimal auszulasten.

Protokolle zum Zugriff auf Server-Ressourcen unter Microsoft Windows Betriebssystemen nach dem Prinzip der entfernten Präsentation wurden erstmals ca. 1995 implementiert. Seither hat das damit auch unter Windows mögliche Server Based Computing (SBC) mit Terminal Servern und im Nachgang auch virtuellen Desktops eine technische Reife erlangt, die dem Betriebsmodell der verteilten Ausführung und Datenhaltung mittels konventioneller Personal Computer nicht nachsteht. Energie- und ressourcensparende Thin Clients haben sich entsprechend als Alternative zu herkömmlichen Arbeitsplatz-Computern und ihrer lokalen Datenverarbeitung etabliert. Die Leistungsfähigkeit der Thin Clients hängt jedoch maßgeblich von der Kapazität der Server-Infrastruktur im Rechenzentrum ab.

Die vorliegende Dissertation greift dieses Thema auf und entwirft ein Referenzmodell für das Kapazitätsmanagement von Server Based Computing Infrastrukturen mit dem Ziel, vorhandene wie auch neu zu konzipierende Systeme zu planen und in einem iterativen Prozess weiterzuentwickeln. Der zu Grunde liegende Ansatz baut auf Methoden und Sprachen der Referenzmodellierung auf. Zunächst wird die aus fünf Schichten bestehende Gesamtsicht einer Server Based Computing Infrastruktur entworfen. Aus diesem Referenzmodell werden nach einem methodischen Vorgehen konkretere Informationsmodelle abgeleitet und in der Sprache der Fundamental Modeling Concepts (FMC) notiert. Ein solches Modell kann anschließend im Rahmen einer Simulation oder einer analytischen Herangehensweise dazu verwendet werden, bereits bei der Konzeption verschiedene Handlungsalternativen zu untersuchen und bezüglich der Kapazität der Ressourcen zu bewerten.

Das Referenzmodell und seine Methodik werden anhand eines exemplarischen Szenarios mit verschiedenen Gruppen von Anwendern und Arbeitsplatzgeräten auf der Client-Seite sowie mehreren Profilen von Anwendungen auf der Server-Seite erprobt. Hierbei wird deutlich, dass die modellbasierte Herangehensweise einen wertvollen Beitrag zum Kapazitätsmanagement leisten kann, ohne dass vorab der tatsächliche Aufbau einer neuen IT-Infrastruktur durch die Installation eines physischen Prototypen und die Simulation von Arbeitslasten darauf notwendig wäre.

Abstract

A worldwide rapidly increasing need for assistance of staff by powerful IT-systems leads to an equally ever growing demand for technologies that enable organizations to provide desktop environments and applications to their end users in an efficient and effective way. In terms of both ecologic and economic aspects, the deduced requirement is to size existing hardware and software platforms as suitable as possible for present and future needs, and to allow for an optimum utilization of the system capacities.

Access protocols on server resources based on Microsoft Windows operating systems within the scope of remote presentation were implemented for the first time around 1995. Since then, Server Based Computing (SBC), with terminal servers and virtual desktops later on, has reached a technical maturity which is not inferior to the distributed issue of the operating modeland data storage as used in conventional personal computers. Accordingly, energy and resource saving thin clients have established themselves as an alternative to conventional desktop computers and local data processing. Their performance, however, depends significantly on the capacity of the server infrastructure located in the data center.

The present thesis takes up this subject and outlines a reference model for the capacity management of Server Based Computing infrastructures with the intention to plan novel designed systems and, further, to develop both these as well as exsisting ones by means of an iterative process.

The underlying approach bases upon methods for reference modeling and languages. Initially, a global view of a Server Based Computing infrastructure consisting of five layers is developed. From this reference model, more precise information models are derived following a methodological approach and are stated according to language elements of the Fundamental Modeling Concepts (FMC).

Such model can be used subsequently within the scope of a simulation or an analytical approach, hereby aiming to investigate and evaluate various alternative courses of action regarding the capacity of resources already during the conception phase.

The reference model and its methodology are evaluated using an exemplary scenario with different groups of users and workstation devices on the client side and several profiles of applications on the server side. This shows clearly that the model-based approach can make a valuable contribution to the capacity management, without requiring the actual implementation of a new IT infrastructure by building a physical prototype and simulating workloads within this prototype.

Inhalt

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Das im Titel der Arbeit erwähnte *Server Based Computing (SBC)* ist ein Betriebsmodell, dessen Anwendung im Bereich der IT-Systemtechnik dazu dient, Anwender der Systeme mit Desktop-Umgebungen und Applikationen zu versorgen. Die vorliegende Dissertation adressiert dieses Thema mit dem Fokus auf der Entwicklung *eines Referenzmodells für Server Based Computing Infrastrukturen und dessen Anwendung für das Kapazitätsmanagement*. Dieses Referenzmodell und die Ableitung von Informationsmodellen daraus bilden den wissenschaftlichen Beitrag der Arbeit und sind für die Planung und Weiterentwicklung einer solchen SBC Infrastruktur von Nutzen. Das folgende Kapitel 1.1 beschreibt zunächst die gegenwärtige Ausgangssituation, die sodann im Kapitel 1.2 aufgegriffen wird und zur Ableitung des zu erreichenden Ziels führt. Das Vorgehen zur Beantwortung der daraus resultierenden Fragen und der Aufbau des Dokuments werden anschließend in Kapitel 1.3 dargelegt.

1.1 Ausgangssituation

Die Architektur von Systemlandschaften für den Betrieb von Desktops und Anwendungen hat sich in den vergangenen Jahrzehnten signifikant gewandelt. Die vormals stationäre Nutzung, primär an fixen Büroarbeitsplätzen in den Liegenschaften eines Unternehmens, hat sich verändert zu flexiblen Arbeitsweisen an unterschiedlichen, wechselnden Standorten und ebenso unterschiedlichen, wechselnden Endgeräten.

Insbesondere vor dem Hintergrund dieser zunehmend mobileren Arbeitswelten – mit Nutzung von IT-Ressourcen auch von zu Hause, unterwegs sowie an national und international verteilten Unternehmensstandorten – steigt der Bedarf an Technologien, die es ermöglichen, Applikationen effizient und gleichzeitig im Einklang mit Anforderungen der IT-Sicherheit bereitzustellen.

Eine naheliegende Konsequenz ist die zunehmende Verbreitung von Notebooks, die klassische Desktop-Computer verdrängen und das Arbeiten nach dem herkömmlichen Prinzip der verteilten Datenhaltung und -verarbeitung ermöglichen. Einem gegenläufigen Trend zufolge erlauben Unternehmen ihren Endanwendern jedoch zunehmend auch die Nutzung privater Endgeräte, was unter dem Begriff „Bring Your Own Device (BYOD)“ Verbreitung findet. Nach Untersuchungen des Marktforschungsunternehmens Gartner (vgl. [Gartner, 2013]) wird sich dieser Trend in den kommenden Jahren nicht nur beschleunigen, sondern auch einen Paradigmenwechsel einleiten, so dass viele Unternehmen die Nutzung privater Geräte nicht nur *erlauben*, sondern sogar *verlangen*.

Damit verbunden sind allerdings auch Bedenken bezüglich der Sicherheit der Daten. Diese führen dazu, dass das Server Based Computing in BYOD-Szenarien verstärkte Anwendung findet vor dem Hintergrund, dass in diesem Fall die Datenverarbeitung komplett zentral im

sicheren Rechenzentrum des Unternehmens stattfindet und keine sensiblen Daten auf potenziell unsichere Endgeräte heruntergeladen werden bzw. dort verbleiben.

Neben der IT-Sicherheit tragen Anforderungen hinsichtlich der Kompatibilität von Anwendungen ebenso zu diesem Trend bei, da gerade im Unternehmenseinsatz viele Anwendungen auf Basis von Microsoft Windows existieren, die nicht oder nur mit enormem Aufwand auf andere Plattformen portiert werden können. Dem steht die – auch auf Grund von BYOD-Ansätzen – wachsende Heterogenität der Client-Landschaften gegenüber, da alternative Endgeräte zunehmende Verbreitung finden.

Exemplarisch erwähnt seien hier Tablet-Computer unter Betriebssystemen wie Google Android oder Apple iOS, die technisch bedingt überhaupt nicht in der Lage sind, Windows Applikationen auszuführen. Des Weiteren sind ggf. auch Notebooks mit dem Betriebssystem Apple Mac OS X zu unterstützen, auf denen der lokale Betrieb von Windows Anwendungen nur mittels Installation eines weiteren Betriebssystems oder zusätzlicher Virtualisierungssoftware möglich wäre. Auch vor diesem Hintergrund ist das Server Based Computing ein wichtiger Baustein, um die Nutzung des existierenden Altbestands an Applikationen auch von abweichenden Plattformen und Endgerätetypen aus zu ermöglichen.

Ein weiterer Aspekt, der den zunehmenden Einsatz von Technologien des Server Based Computings erfordert, ist der wachsende Markt von Thin Client Arbeitsplatzcomputern. Wenngleich diese Geräte in Relation zu herkömmlichen Personal Computern und Notebooks immer noch ein Nischenprodukt darstellen, liegt ihr Anteil an der Menge der geschäftlich genutzten Clients nach Erkenntnissen der Marktanalysten der International Data Corporation (IDC) bereits bei über vier Prozent mit weiter steigender Tendenz[1].

Da diese Thin Clients funktional abhängig von Terminal Servern und virtuellen Desktops im Rechenzentrum sind, bedeuten steigende Verkaufszahlen von Thin Clients ebenso steigende Anforderungen an die Kapazität der Server Based Computing Infrastrukturen in den Unternehmen.

1.2 Zielsetzung und Aufgabenstellung

Vor dem dargelegten Hintergrund besteht die Zielsetzung der vorliegenden Dissertation darin, ein Referenzmodell für die Planung der Server Based Computing Infrastrukturen im Hinblick auf deren Leistungsfähigkeit bereitzustellen. Dieses Modell soll eine Kapazitätsprognose für Infrastrukturen ermöglichen, ohne dass vorab der tatsächliche Aufbau eines physischen Prototyps des gewünschten Zielsystems und die Ausführung von realen Arbeitslasten darauf notwendig wären.

[1] IDC Pressemeldung: „Thin Clients Emerge Unscathed from the Battle with PCs in EMEA" http://www.idc.com/getdoc.jsp?containerId=prCZ25346814 (abgerufen am 01.02.2016).

Der hier vorgestellte Grundgedanke basiert auf Ansätzen der Referenzmodellierung, die eine hersteller- und produktneutrale Gesamtsicht auf SBC Infrastrukturen schaffen, aus der sich konkretere Informationsmodelle für einen jeweiligen Anwendungsfall ableiten lassen.

Der Forschungsbeitrag, den dieses Dokument leistet, liegt in der grundsätzlichen Konzeption und Entwicklung des Referenzmodells nebst seiner Methodik und Sprachwelt sowie dessen Einsatz im Bereich der IT-Systemtechnik, für den ein solches strukturiertes Modell bislang nicht existierte. Der Einsatz des Referenzmodells wird anhand eines exemplarischen Szenarios erprobt und validiert. In diesem Zusammenhang werden die folgenden zentralen Forschungsfragen zum Gegenstand der Untersuchung:

> *Kann die Referenzmodellierung helfen, das Kapazitätsmanagement im Bereich des Server Based Computings zu vereinfachen und zu strukturieren? Wie muss ein solches Modell ausgestaltet werden?*

Diese Fragen widmen sich dem Aspekt, dass bislang zu wenige Ansätze zur Planungsunterstützung existieren, die ohne den Prototypenbau und die Ausführung realer Arbeitslasten auskommen. Die Simulation auf einem Prototypen ist jedoch gerade für Betreiber von sehr kleinen bis mittelgroßen Umgebungen kein gangbarer Weg, da nicht in ausreichendem Maße finanzielle Mittel und Ressourcen zur Verfügung stehen, um diese Vorgehensweise zu verfolgen. Somit stellt sich die Frage, ob der modellbasierte Ansatz helfen kann, solche Aufwände zu reduzieren oder zu vermeiden.

Wird nun als Arbeitshypothese angenommen, dass die Referenzmodellierung tatsächlich ein valider Ansatz zur Unterstützung des Kapazitätsmanagements ist, so lassen sich weitere Forschungsfragen zur inhaltlichen Ausgestaltung des Referenzmodells ableiten:

> *Wie lassen sich aus dem Referenzmodell konkretere Informationsmodelle neu zu schaffender oder bereits existierender Server Based Computing Infrastrukturen ableiten? Und wie kann ein solches Informationsmodell im praktischen Einsatz von Nutzen sein?*

Letztere Frage bezieht sich auf die weitere Verwendung eines Informationsmodells und lässt sich präzisieren zu:

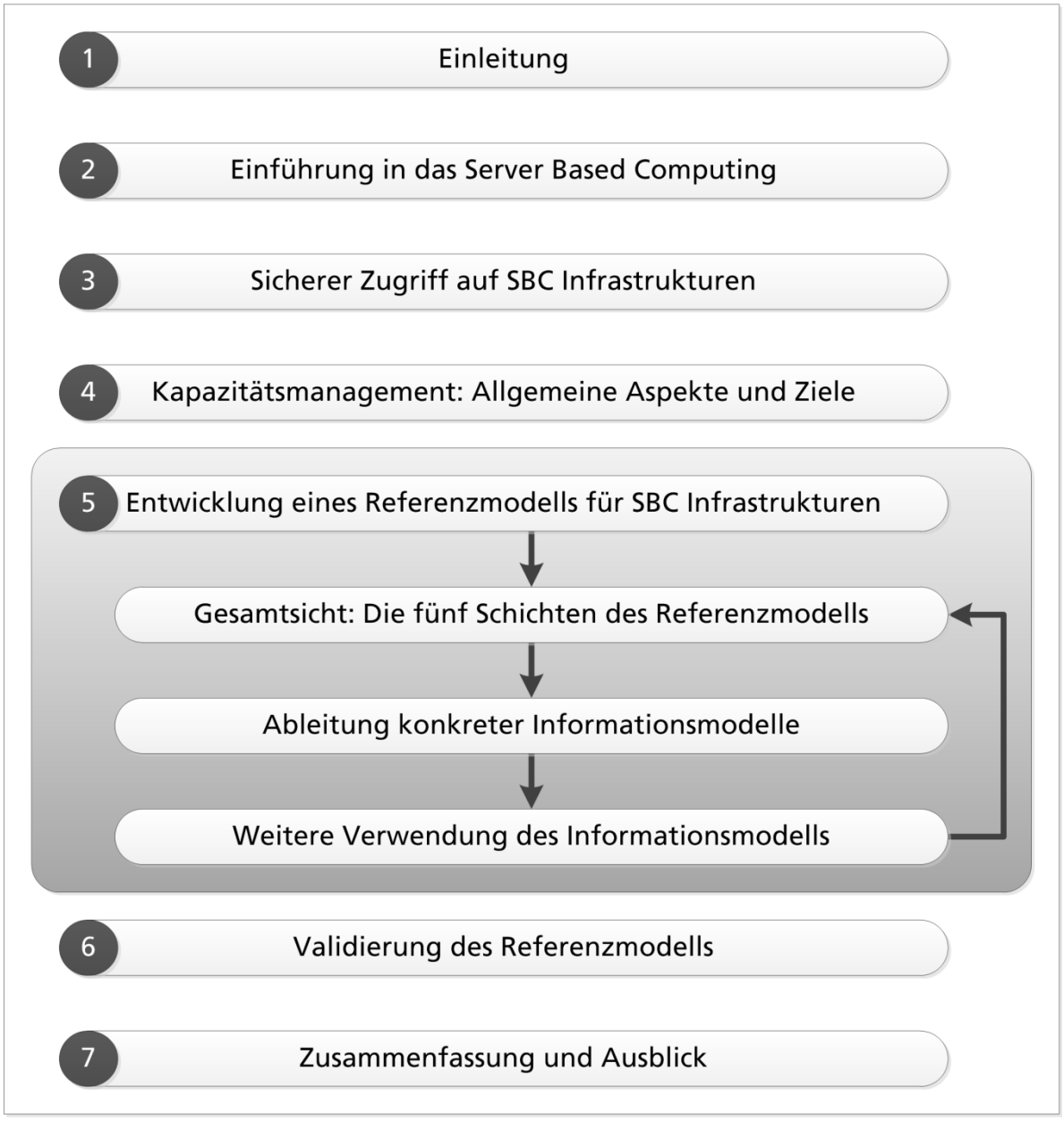
> *Wie lässt sich ein Informationsmodell in ein quantitativ bewertbares analytisches Modell überführen, das konkrete Aussagen im Hinblick auf die von einer SBC Infrastruktur zu erwartende Kapazität und Leistungsfähigkeit ermöglicht?*

Für die Zielsetzung dieser Arbeit folgt daraus, einen Ansatz zu erarbeiten, der Administratoren und IT-Systemtechniker im Bereich des Server Based Computings bei der Analyse, Bewertung und Weiterentwicklung ihrer Systemlandschaften unterstützt.

1.3 Aufbau der Arbeit

In Kapitel 2 werden die wesentlichen Grundlagen und der Stand der Technik in Bezug auf das Server Based Computing vorgestellt. Die Schwerpunkte liegen hierbei auf dem Prinzip der entfernten Präsentation, insbesondere dessen historischer Entwicklung und Implementierung unter Microsoft Windows Betriebssystemen. Weiterhin werden relevante Technologien der Server- und Desktop-Virtualisierung sowie Ausprägungen von zu deren Betrieb notwendiger Client- und Server-Hardware erörtert.

Abbildung 1-1: Aufbau der Arbeit



Quelle: Eigene Darstellung

Kapitel 3 erweitert dann den Fokus auf Technologien für den sicheren Zugriff auf Server Based Computing Infrastrukturen über öffentliche und damit unsichere Netze. Weitere Aspekte der IT-Sicherheit werden dargelegt, soweit sie Bezug zu Terminal Servern sowie Desktop-Virtualisierung haben und dem tieferen Verständnis der im Folgenden betrachteten Infrastruktur dienlich sind. Generelle Aspekte und Ziele des Kapazitätsmanagements werden in Kapitel 4 beschrieben.

Das Kapitel 5 beinhaltet die Lösungsbausteine des eigenen Ansatzes und somit den Kern der Entwicklung *eines Referenzmodells für Server Based Computing Infrastrukturen* mitsamt einer Methodik zu *dessen Anwendung für das Kapazitätsmanagement*, wobei letztere die logische Abfolge der nötigen Arbeitsschritte beschreibt. Bei einer kontinuierlichen Verwendung sind diese Arbeitsschritte entsprechend zyklisch zu sehen (Abbildung 1-1).

Die Kapitel 5.1 bis 5.3 geben zunächst eine allgemeine Einführung in den Begriff der Referenzmodellierung, seine Einordnung in den Kontext sowie Referenzmodellierungs-methoden und -sprachen. Kapitel 5.4 ist daran anschließend der Entwicklung der Gesamtsicht des Referenzmodells gewidmet, welches eine SBC Infrastruktur in fünf Schichten untergliedert. Aus dieser Gesamtsicht werden sodann konkretere Informationsmodelle abgeleitet, die letztendlich im Rahmen des Kapazitätsmanagements von SBC Infrastrukturen zum Einsatz kommen. Ein möglicher Ansatz hierzu, basierend auf geschlossenen Warteschlangennetzen und deren algorithmischer Lösung, und dessen Abbildung in der Software VITO werden umfassend beschrieben. Wird eine Umgebung schließlich gemäß den Erkenntnissen aus der Referenzmodellierung tatsächlich implementiert und produktiv eingesetzt, so lässt sich das Referenzmodell mittels Auswertung von Monitoringdaten validieren.

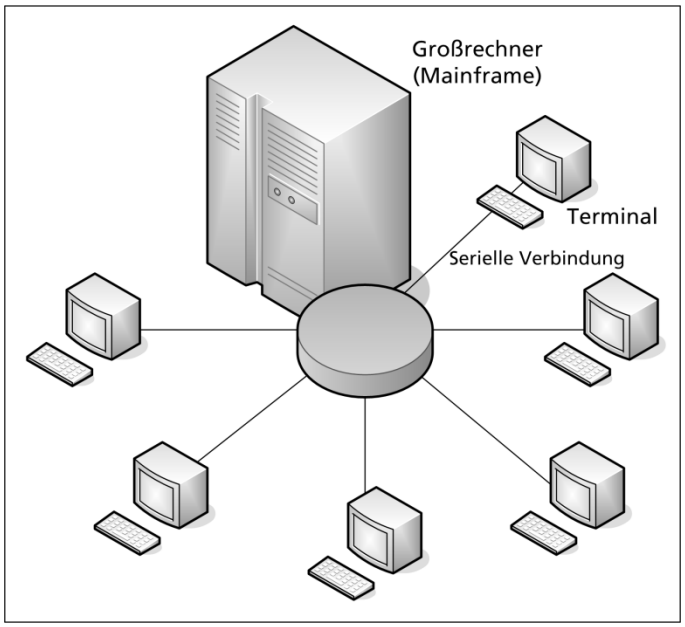
Kapitel 6 widmet sich entsprechend dem Einsatz des Referenzmodells anhand eines konkreten Beispielszenarios und seiner Validierung. Kapitel 7 fasst abschließend die Ergebnisse und Erkenntnisse der Arbeit zusammen und gibt einen Ausblick auf darauf aufbauende, zukünftig zu bearbeitende Forschungsfragen sowie auf die technologischen Entwicklungstrends im Bereich des Server Based Computings und daraus folgende Neuerungen im Bereich von Monitoring und Statistik.

2 Einführung in das Server Based Computing

2.1 Das Prinzip der entfernten Präsentation

Das Funktionsprinzip des Server Based Computings (SBC) basiert auf einem Betriebsmodell, wie es für Großrechner bereits seit den 50er- und 60er-Jahren des vergangenen Jahrhunderts üblich war. Damals war die verfügbare Hardware in Bezug auf die gebotene Leistung und im Vergleich zu den heutigen technischen Möglichkeiten exorbitant teurer. Entsprechend wurden die verfügbaren Rechenkapazitäten in Großrechnern, den sogenannten Mainframes[2], konzentriert. Diese Systeme waren bereits multitaskingfähig und von mehreren Benutzern gleichzeitig verwendbar. Den Zugriff ermöglichten anfänglich einfache Text-Terminals. Einer der bekanntesten Vertreter dieser Gerätegattung war das Terminal „VT100“, entwickelt von der Digital Equipment Corporation (DEC) in den 1970er Jahren[3]. Typisch für dieses und vergleichbare Geräte war, dass diese Clients über serielle Leitungen mit einem Großrechner als Server sternförmig vernetzt wurden (Abbildung 2-1).

Abbildung 2-1: Server Based Computing



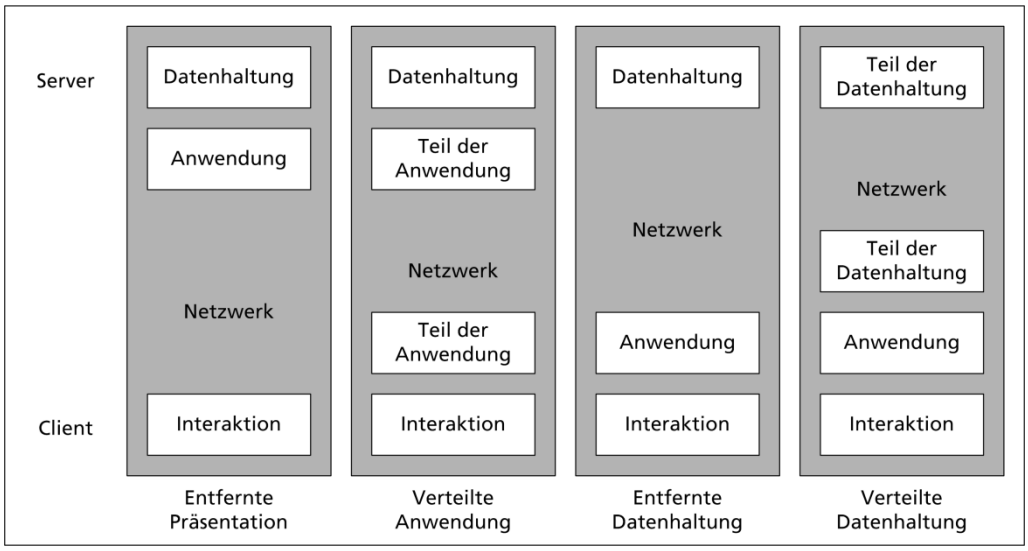
Quelle: Eigene Darstellung

[2] Mainframes sind auch heute noch verbreitet, insbesondere im Bereich von Banken und Versicherungen. Für den Zugriff kommen aber heutzutage in der Regel keine der damaligen Hardware-Terminals mehr zum Einsatz, sondern eine als Software realisierte Terminalemulation, welche auf einem PC oder Thin Client (Kapitel 2.4.1) ausgeführt wird.

[3] Video Terminal: http://vt100.net (abgerufen am 01.02.2016).

Die Netzwerkverbindung diente der Übermittlung von Eingabeinformationen vom Client zum Server und der Übertragung der textbasierten Ausgabe in der entgegengesetzten Richtung. Damit entsprach dieser Aufbau in seiner Architektur dem Prinzip der „entfernten Präsentation“ (Abbildung 2-2), bei der auf den Client lediglich die Interaktion mit dem Anwender entfällt, während Anwendungslogik und Datenhaltung komplett auf Seiten des Servers platziert sind.

Abbildung 2-2: Die unterschiedlichen Betriebsmodelle



Quelle: Eigene Darstellung, nach [Tritsch, 2003], Seite 7

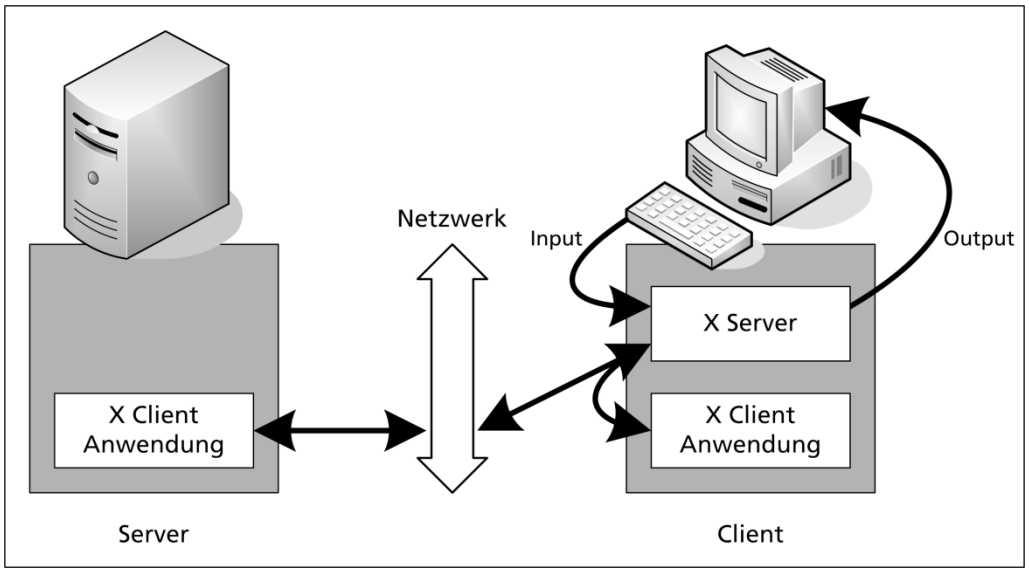
Alternativ bietet sich eine zunehmende Verlagerung von Ressourcen in Richtung des Clients an, wobei „verteilte Anwendungen“ die Anwendungslogik teilweise auf den Client verlegen. Nach dem Prinzip der „entfernten Datenhaltung“ findet die Datenverarbeitung dagegen vollständig auf dem Client statt und nur die Speicherung obliegt noch dem Server, während im Fall der „verteilten Datenhaltung“ auch die Speicherung der Daten teilweise oder gar ganz dem Client zufällt. (vgl. [Tritsch, 2003], Seite 6 ff.). Für die Realisierung dieser Betriebsmodelle war aber zunächst die Entwicklung leistungsfähigerer Netzwerkinfrastrukturen erforderlich, welche die Möglichkeiten seriell angebundener Terminals übertrafen.

2.1.1 Unix-artige Betriebssysteme und das X Window System

Die weitere Entwicklung führte ab 1969 zum Uniplexed Information and Computing Service (UNIX) und zahlreichen weiteren Unix-artigen Betriebssystemen. Ebenso wie die Mainframes waren auch diese Systeme darauf ausgelegt, im Multitasking- und Multiuser-Betrieb zu arbeiten, so dass sich mehrere Benutzer einen Server teilen konnten. Während die Entwicklung zur heute dominierenden TCP/IP-Protokollfamilie und zu weiteren Netzwerkprotokollen erstmals die entfernte und verteilte Datenhaltung in größerem Umfang erlaubte, blieb das Konzept der entfernten Präsentation als integraler Bestandteil der Unix-artigen Betriebssysteme

dennoch erhalten und wurde über die ursprünglich rein textbasierten Zugriffe noch deutlich erweitert. So entstand Mitte der 1980er Jahre am Massachusetts Institute of Technology (MIT) das X Window System[4], welches auf einer Vielzahl von Unix-Derivaten und unixoiden Betriebssystemen implementiert wurde und das verteilte Arbeiten mit grafischen Benutzeroberflächen erlaubt. Das X Window System, oder kurz X, dient der Darstellung von Bitmap-basierten Grafikinhalten. Im Kontext von X ist die Verwendung der Begrifflichkeiten Client und Server gegenüber der allgemein gebräuchlichen Verwendung gewöhnungsbedürftig. X Client Anwendungen werden auf dem Servercomputer ausgeführt, während der X Server auf dem Clientcomputer läuft, wo er Grafikausgabe und Eingabegeräte verwaltet (vgl. [Nye, 1994], Kap. 1.2 ff.). Dieses Konzept kommt unabhängig davon zum Einsatz, ob eine Anwendung lokal oder auf einem entfernten System ausgeführt wird. Dies bedeutet, dass ein lokal laufender X Server sowohl Anzeigeinformationen von ebenfalls lokal laufenden X Client Anwendungen als auch von X Client Anwendungen auf entfernten Computern empfangen und darstellen kann (Abbildung 2-3).

Abbildung 2-3: X Window System



Quelle: Eigene Darstellung, nach [Nye, 1994], Kap. 2.1

Die Kombination dieser Informationen aus unterschiedlichen Quellen zu einer Benutzeroberfläche obliegt nicht dem X Window System, da dieses nur die grundlegenden Funktionen zum Zeichnen von Punkten bzw. geometrischen Formen umfasst sowie „ein Netzwerkprotokoll, das es ermöglicht, ein X-Programm auf Rechner A auszuführen und die Ergebnisse (via Netzwerk) auf Rechner B darzustellen“ (vgl. [Kofler, 1999], Seite 281). Die Darstellung von Dialogen, Fenstern und das optische Gesamterscheinungsbild der Benutzeroberfläche regelt ein sogenannter Window Manager. Dabei handelt es sich um eine

[4] X Window System, http://www.x.org (abgerufen am 01.02.2016).

privilegierte X Anwendung, die typische Aufgaben wie das Verschieben und Skalieren von Fenstern übernimmt und die Ausgaben von lokalen und entfernten X Anwendungen aggregiert (vgl. [Nye, 1994], Kap. 1.2.3). Nachdem in der Vergangenheit zahlreiche verschiedene Window Manager, wie Motif, OpenLook und Common Desktop Environment (CDE) (vgl. [Kofler, 1999], Seite 281 f.), miteinander konkurrierten, haben sich in der jüngeren Vergangenheit mit der wachsenden Verbreitung des unixoiden Betriebssystems Linux vor allem die Window Manager GNU Network Object Model Environment (GNOME)[5], K Desktop Environment (KDE)[6] sowie XForms Common Environment (Xfce)[7] durchgesetzt. All diesen Window Managern ist gemeinsam, dass sie mit dem X Window System als Basis von Grund auf dafür konzipiert sind, wahlweise nach dem Betriebsmodell der „entfernten Datenhaltung", der „verteilten Anwendungen/Datenhaltung" oder auch der „entfernten Präsentation" zu arbeiten. In letzterem Fall dient der lokale Computer, ebenso wie das seriell angebundene Terminal als historisches Vorbild, nur der Grafikausgabe und der Interaktion mit dem Benutzer. Das Prinzip des Server Based Computings ist damit in der Entwicklungsgeschichte der Unix-Derivate und unixoiden Betriebssysteme als grundlegendes Betriebskonzept fest verankert und nicht als Spezialfall anzusehen – ganz im Gegensatz zur Entwicklung der Microsoft Windows Betriebssystemfamilie.

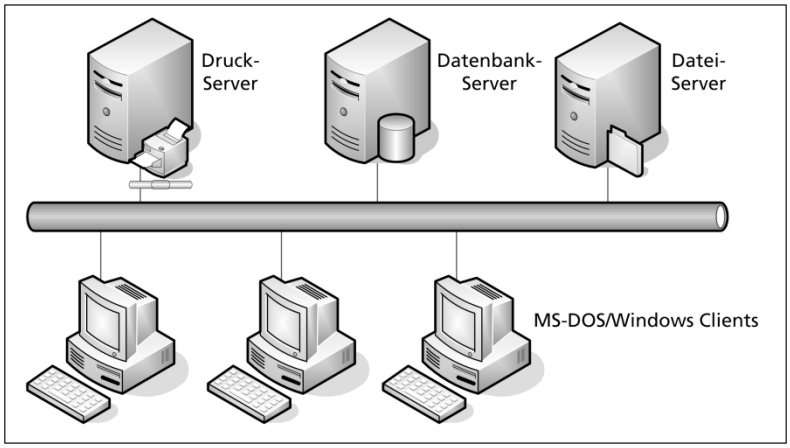
2.1.2 Microsoft Windows

Die rasante Verbreitung des IBM PC und kompatibler Systeme zu Beginn der 1980er Jahre und damit einhergehend der wirtschaftliche Erfolg der Firma Microsoft mit ihrem Betriebssystem MS-DOS und der grafischen Benutzeroberfläche Windows begründeten sich vor allem in der Tatsache, dass damit ein standardisiertes Massenprodukt zu einem Preis, der bislang im Zusammenhang mit IT-Komponenten nicht vorstellbar war, verfügbar wurde. So gelang es, mit den Windows-Versionen ab 3.0 und insbesondere dem für den Netzwerkeinsatz optimierten Windows 3.11 for Workgroups (WfW), eine breite Benutzerbasis zu erschließen. Dies ermöglichte die Entwicklung hin zu einem dezentralen Netzwerk nach den Betriebsmodellen der verteilten Anwendungen und Datenhaltung (Abbildung 2-4).

Der Durchbruch als akzeptierte Plattform im geschäftlichen Alltag blieb dem System dennoch zunächst verwehrt, da das darin realisierte kooperative Multitasking nicht für den stabilen, parallelen Betrieb mehrerer geschäftskritischer Anwendungen geeignet war. Denn beim kooperativen Multitasking obliegt die Verwaltung der Ressourcen wie Prozessor und Hauptspeicher den Anwendungsprogrammen. Dies bedeutet, dass eine Anwendung selbständig genutzte Ressourcen für andere Anwendungen freigeben muss, was dazu führen kann, dass eine fehlerhafte Anwendung die Maschine blockiert. Beim präemptiven Multitasking verwaltet dagegen das Betriebssystem selbst die Ressourcen, kann somit die Freigabe derselben erzwingen und fehlerhafte Anwendungen suspendieren oder terminieren.

[5] GNU Network Object Model Environment (GNOME): http://www.gnome.org (abgerufen am 01.02.2016).
[6] K Desktop Environment: http://www.kde.org (abgerufen am 01.02.2016).
[7] Xfce Desktop Environmend: http://www.xfce.org (abgerufen am 01.02.2016).

Abbildung 2-4: Verteilte Anwendungen/Datenhaltung



Quelle: Eigene Darstellung

Die Einschränkungen der auf MS-DOS basierenden Windows-Benutzeroberfläche ließen Microsoft bereits zum Ende der 1980er Jahre mit der Entwicklung eines komplett neu konzipierten 32-Bit Betriebssystems unter dem Projektnamen „Windows New Technology" beginnen. Das daraus hervorgegangene Windows NT wurde schließlich im Jahr 1993 veröffentlicht und startete, um die Verwandtschaft zum gleichzeitig verfügbaren Windows als Aufsatz von MS-DOS zu verdeutlichen, direkt mit der Versionsnummer 3.1 (vgl. [Tritsch, 2003], Seite 3). In diesem neuen Betriebssystem war zwar präemptives Multitasking realisiert (vgl. [Dapper, 1997], Seite 15 ff.), weiterhin fehlten aber Möglichkeiten zur Nutzung von Anwendungen nach dem Prinzip der entfernten Präsentation sowie die Multi-User-Unterstützung – für Nutzer von Unix-artigen Betriebssystemen seit Jahren eine Selbstverständlichkeit.

2.1.2.1 Windows NT Terminal Server Edition

Microsoft gestattete daher im Rahmen eines Lizenzabkommens der Firma Citrix Systems den Zugriff auf den Quellcode von Windows NT, um eine entsprechende Funktionalität als Erweiterung für das Betriebssystem zu entwickeln. Diese kam in Form des Produkts WinFrame 1995 auf den Markt[8] und erlaubte mehreren Benutzern das interaktive Arbeiten auf Windows Servern im Rahmen der entfernten Präsentation.

Dies wurde realisiert durch die von Citrix geschaffene Erweiterung namens MultiWin, die den Multi-User-Betrieb auf Windows NT implementierte. Der Fernzugriff wurde möglich durch das Citrix eigene Protokoll „Independent Computing Architecture (ICA)", ein proprietäres Protokoll auf den Ebenen 4-7 des ISO/OSI-Referenzmodells, das ähnlich dem Funktionsprinzip des X Window Systems die Darstellung von Windows Dialogen und Fenstern auf entfernten Clientcomputern ermöglichte (vgl. Kapitel 2.1.3.4). Das Produkt brachte einen derart

[8] Vgl. „20 Years of Citrix History" unter http://www.citrix.com/go/20.html (abgerufen am 01.02.2016).

unerwarteten Erfolg mit sich, dass Microsoft die zu Grunde liegende Technologie MultiWin lizenzierte und mit der nächsten Version 4.0 von Windows NT unter dem Entwicklungsnamen „Hydra" ein eigenes Produkt, die Windows NT 4.0 Terminal Server Edition (TSE), entwickelte. Gegenstand des Lizenzabkommens war allerdings nur die Mehrbenutzerumgebung MultiWin und nicht das ICA Protokoll, so dass Microsoft mit dem Remote Desktop Protocol (RDP) ein eigenes Protokoll zur Realisierung der entfernten Präsentation entwickelte (vgl. Kapitel 2.1.2.2). War die Windows NT 4.0 TSE noch ein komplett eigenständiges Produkt mit separater Code-Basis und somit auch inkompatibel mit den regulären Windows NT Service Packs, sind die Terminaldienste seit Windows 2000 eine feste Komponente des Betriebssystems (vgl. [Mathers, 2000], Seite 27-29).

Der stabile Betrieb der Terminaldienste basiert auf dem Prinzip, das Betriebssystem selbst in einem geschützten Bereich, dem Kernelmodus, ablaufen zu lassen. Während die Benutzersitzungen zwar zwecks Zuteilung und Verwaltung von Ressourcen mit den Terminaldiensten im Kernelmodus interagieren, laufen die eigentlichen Anwendungen der Endbenutzer im weniger privilegierten Benutzermodus ohne die Möglichkeit, durch direkte Zugriffe auf Systemkomponenten oder gar Hardware das System zu beeinträchtigen (vgl. [Tritsch, 2003], Seite 12 ff.).

2.1.2.2 Remote Desktop Protocol (RDP)

Da das ICA Protokoll nicht Bestandteil der Lizenzvereinbarung zwischen Microsoft und Citrix war, wurde es im Rahmen der Entwicklung der Windows NT 4.0 TSE erforderlich, ein eigenes Übertragungsprotokoll zu implementieren, um das Prinzip der entfernten Präsentation in Windows zu realisieren. Microsoft entwickelte dazu das proprietäre Remote Desktop Protocol (RDP), welches als verbindungsorientiertes Protokoll ebenso wie ICA auf den Ebenen 4-7 des ISO/OSI-Referenzmodells residiert.

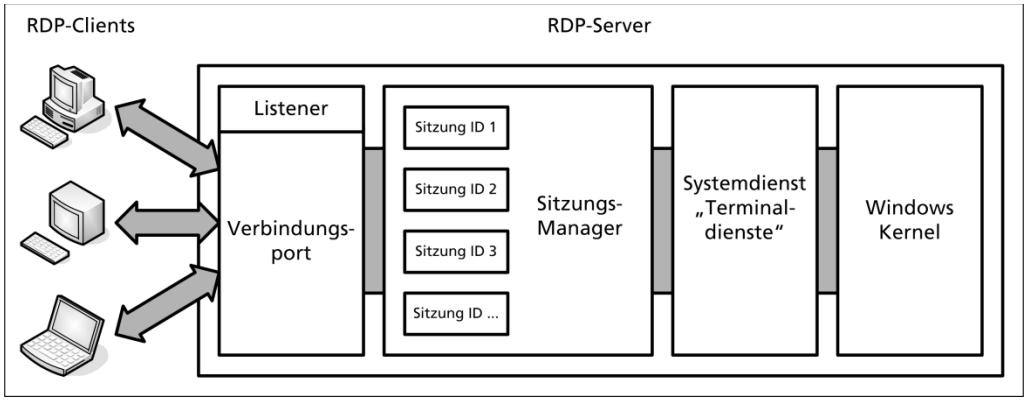
RDP basiert auf den Empfehlungen T.125[9] und T.128[10] der International Telecommunication Union (ITU). Die Funktionsweise ähnelt dem X Window System, wobei im Kontext von RDP die Begriffe Client und Server in der üblichen Weise verwendet werden. Auf dem RDP-Server kann pro Netzwerkkarte eine virtuelle RDP-Verbindung für ein Netzwerkprotokoll und einen Port erstellt werden (Abbildung 2-5). In der Regel findet hierfür der TCP-Port 3389 Verwendung. Auf diesem Port wartet der RDP-Listener auf Verbindungsanfragen von Clients. Wenn ein RDP-Client den RDP-Server kontaktiert, erstellt der Sitzungsmanager eine Sitzung, d. h. einen

[9] ITU-T T.125 „Multipoint communication service protocol specification": Die Empfehlung beschreibt ein Protokoll für einen Kommunikationsdienst, der wiederum in der ITU-T T.122 „Multipoint communication service – Service definition" niedergelegt ist und einen Mechanismus umfasst, um Mehrpunktverbindungen zu entsprechend ausgelegten Applikationen zu ermöglichen (vgl. http://www.itu.int/rec/T-REC-T.122-199802-I/en (abgerufen am 01.02.2016)).

[10] ITU-T T.128 „Multipoint application sharing": Die Empfehlung beschreibt ein Protokoll, das es erlaubt, Applikationen im Kontext einer Sitzung auf einem Computer auszuführen und die Anzeige an anderen Standorte zu veröffentlichen: „...it enables remote viewing and control of a single application instance to provide the illusion that the application is running locally." (vgl. http://www.itu.int/rec/T-REC-T.128-200806-I/en (abgerufen am 01.02.2016)).

virtuellen Desktop, mit einer eindeutigen ID und verbindet den Client mit dieser Sitzung (vgl. [Madden, 2004], S. 34 ff.).

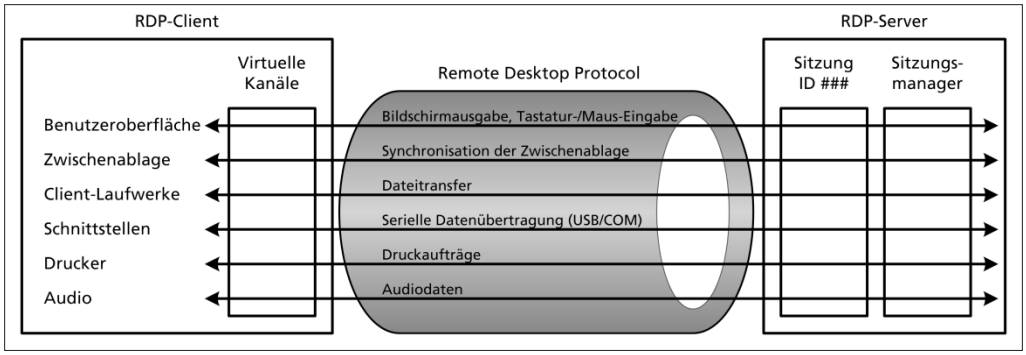
Abbildung 2-5: RDP-Client/-Server



Quelle: Eigene Darstellung, nach [Madden, 2004]

Ist die Verbindung etabliert, nimmt der RDP-Server Tastatur- und Maus-Befehle vom RDP-Client entgegen und sendet die Bildschirmausgabe der Server-seitig ausgeführten Anwendungen an den Client. Dies wird durch Mechanismen wie Kompression und einen Client-seitigen Cache optimiert (vgl. [Tritsch, 2003], S 82 f.). Über sogenannte virtuelle Kanäle ist zudem die Erweiterung des Funktionsumfangs möglich. Im Lauf der Entwicklung des Protokolls haben Microsoft und auch Drittanbieter auf diesem Weg kontinuierlich zusätzliche Funktionen integriert. Diese werden auf dem Client üblicherweise durch das Laden entsprechender Dynamic Link Libraries (DLL) in die RDP-Client-Anwendung realisiert (vgl. [Madden, 2004], S. 40 ff.). Hier sind beispielsweise der Datenaustausch mit lokalen Speichermedien des Clients, Nutzung von lokalen Clientdruckern und Smartcards oder auch die Audioübertragung zu nennen (Abbildung 2-6).

Abbildung 2-6: Virtuelle Kanäle in RDP



Quelle: Eigene Darstellung, nach [Madden, 2004]

Eine RDP-Verbindung unterstützt bis zu 31 solcher virtuellen Kanäle, die bereits beim Verbindungsaufbau durch den Austausch von Protocol Data Units (PDU) statisch zwischen Client und Server ausgehandelt werden und für die Dauer einer Sitzung Bestand haben (vgl.

[Anderson, 2010], Seite 296 ff.). Mit der Einführung der RDP Version 7.0 auf dem Windows Server 2008 wurde dieses Konzept um dynamische virtuelle Kanäle erweitert, die zur Laufzeit einer Sitzung etabliert und wieder entfernt werden können. Auf diese Weise werden zusätzliche Funktionen, wie beispielsweise ein Audio-Rückkanal vom Client zum Server oder der Zugriff auf USB-Geräte, die erst zur Laufzeit der Sitzung eingesteckt werden (Plug & Play (PnP)), möglich. Dies setzt allerdings voraus, dass sowohl Client als auch Server eine aktuelle Version von RDP implementieren, und beschränkt die Verfügbarkeit erweiterter Funktionen damit in der Regel auf ebenso aktuelle Betriebssysteme aus dem Hause Microsoft (vgl. [Anderson, 2010], Seite 306), für die der Hersteller selbst proprietäre RDP Clients anbietet. Darüber hinaus war seitens Microsoft lange Zeit lediglich ein RDP Client für die Plattform Apple Mac OS X verfügbar. Inzwischen stehen neben Mac OS X[11] auch Clients für mobile Endgeräte unter Windows Phone 8, Apple iOS und Google Android zur Verfügung (vgl. [Knermann, 2014a]).

Weitere Plattformen, insbesondere die Unix-artigen Betriebssysteme, werden seitens des Herstellers nicht mit passenden Clientanwendungen versorgt. Grundlegende Informationen zum Verbindungsaufbau und zur entfernten Präsentation von grafischen Inhalten hat Microsoft allerdings offengelegt[12], so dass darauf basierend freie Implementierungen von RDP Clients[13] und auch eines RDP Servers[14] für Unix-artige Betriebssysteme entstehen konnten. Diesen Implementierungen von RDP ist aber gemeinsam, dass sie auf Basisfunktionen beschränkt sind, während die erweiterten Funktionen, wie z. B. die zuvor erwähnte PnP Unterstützung von USB-Geräten, im nicht offengelegten Teil von RDP verankert und somit nur mit den von Microsoft offiziell unterstützten Clientanwendungen vollständig nutzbar sind. Daher sind alternative Ansätze zur Realisierung der entfernten Präsentation auf Windows Servern, allen voran die Lösungen von Citrix Systems, trotz des kontinuierlich wachsenden Funktionsumfangs von Microsoft sehr verbreitet.

2.1.3 Citrix

Wenngleich Microsoft mit der Lizenzierung der MultiWin-Technologie und der Entwicklung von RDP die Multi-User-Unterstützung sowie die Fähigkeit zur entfernten Präsentation direkt in Windows NT und den nachfolgenden Betriebssystemen verankerte, war dies nicht der Schlusspunkt der Zusammenarbeit mit Citrix Systems.

2.1.3.1 Von WinFrame zu XenApp

In der Nachfolge von WinFrame erschien mit Citrix MetaFrame ein Zusatzprodukt zur Erweiterung der Funktionalität der Windows NT TSE (vgl. [Mathers, 2000], S. 68 ff.). In der

[11] http://www.microsoft.com/de-de/download/details.aspx?id=1350 (abgerufen am 01.02.2016).
[12] Remote Desktop Protocol: Basic Connectivity and Graphics Remoting Specification (http://msdn.microsoft.com/en-us/library/cc240445(v=PROT.10).aspx (abgerufen am 01.02.2016)).
[13] Rdesktop (http://www.rdesktop.org), FreeRDP (http://www.freerdp.com (abgerufen am 01.02.2016)).
[14] Der Open Source RDP Server xrdp (http://www.xrdp.org (abgerufen am 01.02.2016)) erlaubt den Zugriff auf einen Desktop des X Window Systems via Remote Desktop Protocol.

Zwischenzeit mehrfach umbenannt – zunächst in Citrix Presentation Server, dann in Citrix XenApp – ergänzt diese Infrastrukturlösung von Citrix die Microsoft Terminaldienste nun bereits in der siebten Generation und optimiert diese für den Einsatz in größeren, heterogenen Unternehmensnetzen[15]. Das Betriebsmodell von Citrix sieht vor, Terminal Server nicht jeweils einzeln als alleinstehende Systeme zu verwalten, sondern diese zu einer sogenannten Server Farm zu verbinden. Die so zusammengeschlossenen Server können über ein gemeinsames Verwaltungswerkzeug von zentraler Stelle aus administriert werden. Die Konfiguration einer Farm mit Informationen über vorhandene Server, Anwendungen und berechtigte Benutzer wird zentral in einer Datenbank abgelegt, die sich wahlweise auf dem ersten Server einer Farm oder auf einem separaten Datenbankserver befinden kann (vgl. [Knermann, 2010]).

2.1.3.2 Von XenApp zu XenDesktop

Bis zur Version 6.5 von XenApp basierte die Administration der Terminal Server Farm auf dem Protokoll „Independent Management Architecture (IMA)“, einer auf der Microsoft Management Console (MMC) basierenden Verwaltungsoberfläche und einer Microsoft SQL Server Datenbank für den Citrix Datastore mit der kompletten Konfiguration der Farm. Mit XenApp 6.5 kam dann die Aufteilung in die Rollen der Controller und Worker hinzu. Erstere bilden den logischen „Kopf“ der Farm und erlauben deren Administration. Die Worker verfügen nur über einen reduzierten IMA-Stack und dienen der Abwicklung der Benutzersitzungen.

Mit diesem Rollenkonzept wurde es vereinfacht, die Worker mittels Citrix Provisioning Services (PVS) aus einem gemeinsamen Basis-Image zu starten (siehe Kapitel 2.3.1) und die Pflege sowie Skalierung einer Farm deutlich zu vereinfachen. Kehrseite der Medaille war, dass die PVS neben eigenen Servern ebenfalls eine eigene Datenbank nebst separatem Verwaltungsprogramm sowie eine Software-Komponente auf den Zielsystemen erforderten.

Sofern Monitoring und statistische Erfassung der Farm gewünscht waren, kam Citrix EdgeSight (siehe Kapitel 12.3 bis 12.5) als zusätzliche Lösung hinzu, wiederum mit eigenen Agenten auf den Zielsystemen, einer zusätzlichen Datenbank und webbasierter Verwaltungsoberfläche. Und das optionale „Power & Capacity Management“ brachte eine weitere Konsole mit sich.

Sollten dann einer solchen Umgebung virtuelle Desktops auf Basis von XenDesktop 5.6 hinzugefügt werden, bedeutete dies, dass neben der XenApp Farm eine eigenständige XenDesktop Site gepflegt werden musste. Um diese Komplexität zweier separater Verwaltungsbereiche aufzulösen, wurden die Entwicklungsstränge von XenApp und XenDesktop ab der Version 7.0 zusammengeführt (siehe Kapitel 2.3.2).

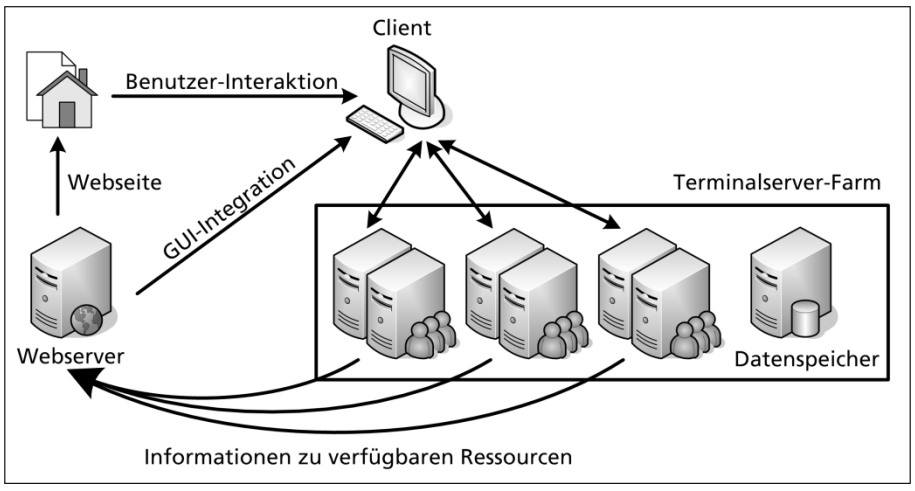
[15] Detaillierte Informationen zur Entwicklung der technischen Funktionen über die verschiedenen Versionen finden sich in [Knermann, 2007], [Knermann, 2009], [Lüdemann, 2009], [Knermann, 2010] sowie [Lüdemann, 2011].

2.1.3.3 Veröffentlichte Anwendungen

Neben der Möglichkeit, innerhalb einer Terminal Server Sitzung auf einen kompletten Windows Desktop zuzugreifen, hat Citrix mit MetaFrame und seinen Nachfolgern das Konzept der veröffentlichten Anwendungen – engl. published applications – geprägt. Dabei wird im Rahmen der entfernten Präsentation keine vollständige Desktop-Umgebung, sondern nur eine einzelne Applikation vom Server an den Client übertragen und dort als sogenanntes „Seamless Window" auf dem lokalen Desktop des Clients angezeigt. Eine Server-seitig bereitgestellte Anwendung integriert sich so nahtlos in die grafische Benutzeroberfläche des Clients und erweckt den Eindruck, als sei es eine lokal installierte und ausgeführte Anwendung.

Microsoft hat mit den neueren Windows Versionen, insbesondere mit dem Windows Server 2008 R2, das Farm-Betriebsmodell für die Remotedesktopdienste[16] adaptiert (vgl. [Knermann, 2008] und [Knermann, 2008a]) und für den Windows Server mit den sogenannten RemoteApps eine Möglichkeit geschaffen, auch ohne Produkte von Drittanbietern Anwendungen nach Art der veröffentlichten Applikationen zu betreiben (vgl. [Knermann, 2011]). Wenngleich die Implementierung seitens Microsoft in Bezug auf Funktionalität und Administrierbarkeit noch hinter Drittanbieter-Lösungen, insbesondere Citrix XenApp, zurückbleibt, weisen die Implementierungen in der grundsätzlichen Funktionsweise Parallelen auf.

Abbildung 2-7: Terminal Server im Farm-Verbund



Quelle: Eigene Darstellung

So nehmen in einem Farm-Verbund die Clients mittelbar über einen Webserver mit der Farm Kontakt auf (Abbildung 2-7). Dieser Webserver ruft von den Terminal Servern Informationen zu den verfügbaren Desktops und veröffentlichten Anwendungen ab und bereitet diese

[16] Mit dem Windows Server 2008 R2 wurden die Terminaldienste in Remotedesktopdienste umbenannt. Der klassische Terminal Server trägt seither die offizielle Bezeichnung Remotedesktop-Sitzungshost – engl. Remote Desktop Session Host (RDSH).

Informationen in Form einer Webseite auf, so dass ein Benutzer interaktiv im Browser die gewünschte Sitzung starten kann. Alternativ können die Sitzungen auch als Verknüpfungen in die Benutzeroberfläche, d. h. im Startmenü oder auf dem Desktop, eines Client-Computers integriert werden, so dass Benutzer diese wie eine lokal installierte Anwendung starten können.

Der Webserver vermittelt daraufhin eine direkte Verbindung zwischen dem Client und einem Server, womit es möglich wird, eine Lastverteilung über die Server der Farm zu erreichen. Anwender müssen sich auf diese Weise nicht mehr darum kümmern, auf welchem Server sich ein gewünschter Desktop oder eine Anwendung befindet, sondern werden automatisch mit dem System mit der geringsten Last verbunden. Der Mechanismus der Lastverteilung erhöht die Fehlertoleranz einer Farm, sofern Server redundant ausgelegt werden. Der Ausfall eines Servers hat zwar den Abbruch der auf diesem System aktiven Benutzersitzungen zur Folge, die Anwender können sich im Fehlerfall aber umgehend an einem der verbliebenen Farm Server neu anmelden. Im Kontext der Infrastrukturlösungen von Citrix kommt dabei standardmäßig nicht RDP, sondern das Protokoll Independent Computing Architecture (ICA) zum Einsatz.

2.1.3.4 Independent Computing Architecture

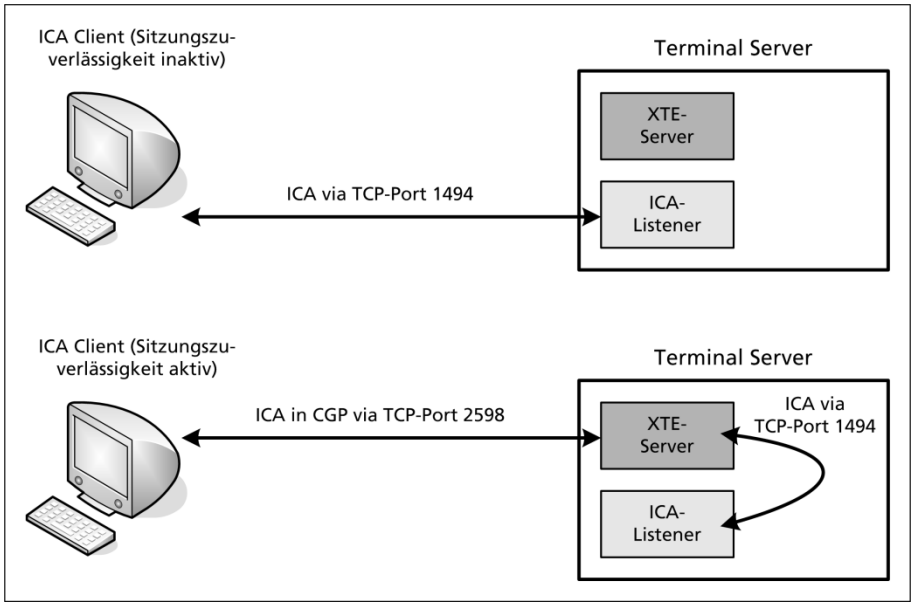
Im Gegensatz zu RDP ist die Spezifikation der Independent Computing Architecture (ICA) nicht, auch nicht in Teilen, offen verfügbar. Entsprechend haben Drittanbieter nicht die Möglichkeit, eigene Client- oder Serverprogramme zu implementieren. Bereits seit WinFrame hatte Citrix das eigene Client-Programm für das ICA Protokoll aber für eine breite Palette an Client-Betriebssystemen angeboten, darunter in der Vergangenheit auch MS-DOS und den Aufsatz Windows 3.x, ältere Versionen von Apple Mac OS sowie diverse Unix-artige Betriebssysteme oder das Symbian OS für mobile Clients. Heute sind Clients, die den aktuellen Funktionsumfang unterstützen, für Microsoft Windows ab der Version Vista, Apple Mac OS X, Linux sowie die Mobil-Plattformen Apple iOS, Google Android, Blackberry und Microsoft Windows Phone verfügbar. Die Implementierung des ICA-Servers ist Teil der Terminal Server Lösung XenApp 6.x für den Microsoft Windows Server 2008 R2[17] sowie der Infrastruktur für virtuelle Desktops XenDesktop (vgl. Kapitel 2.3).

Die grundlegende Funktionalität von ICA ähnelt der von RDP. Auch bei ICA handelt es sich um ein verbindungsorientiertes Protokoll, das auf den Schichten 4-7 des ISO/OSI-Referenzmodells angesiedelt ist und der entfernten Präsentation von Anwendungen und Desktops dient. Ebenso wie RDP kennt ICA das Konzept der virtuellen Kanäle zur Erweiterung der Funktionen. Der ICA Listener arbeitet standardmäßig auf dem TCP-Port 1494. Alternativ gibt es die Möglichkeit, über den Citrx XTE Service den ICA-Datenstrom im sogenannten Common Gateway Protocol (CGP) zu tunneln, um die Funktion der sogenannten Sitzungszuverlässigkeit

[17] Die älteren Betriebssysteme Microsoft Windows Server 2003 (R2) sowie Windows Server 2008 werden nur durch XenApp 5 unterstützt, das nicht mehr weiterentwickelt wird. In der Vergangenheit wurde mit dem Presentation Server bzw. XenApp for Unix 4 auch ein ICA Server für diverse UNIX-Derivate – SUN Solaris 8-10 für die SPARC-Plattform sowie SUN Solaris 10 für die x86 Architektur, IBM AIX 5.1/5.2/5.3 und HP/UX 11/11i – implementiert, der aber ebenfalls nicht mehr weiter gepflegt wird.

zu nutzen (Abbildung 2-8). Dabei wird eine via ICA verbundene Anwendung oder Desktop-Sitzung auf einem Client-Computer im Falle einer Verbindungsstörung für eine definierbare Zeitspanne weiterhin gehalten sowie angezeigt (vgl. [Lüdemann, 2009], S. 134 ff.) und die Kommunikation im Hintergrund dann automatisch wieder aufgenommen. Ohne diese Funktion würde das Fenster der Applikation geschlossen und nach Wiederherstellen der Kommunikation erneut geöffnet, was vom Anwender eher als Funktionsstörung wahrgenommen wird als ein kurzes „Einfrieren" der Sitzung.

Abbildung 2-8: Citrix ICA/CGP Kommunikation



Quelle: Eigene Darstellung, nach [Lüdemann, 2009], S. 135

In jüngerer Vergangenheit hat Citrix unter dem Label „High Definition User Experience (HDX)" das ICA Protokoll um weitere Funktionen ergänzt. „Dies betrifft insbesondere den optimierten Zugriff auf Grafik-lastige Anwendungen und Multimedia. Um insbesondere die Wiedergabe kontinuierlicher Medien zu beschleunigen, wird basierend auf verfügbarer Bandbreite, Rechenleistung des Clients und verfügbaren Codecs ermittelt, ob Inhalte Server- oder Client-seitig gerendert werden. Soll also beispielsweise in einer Remote-Sitzung ein DivX-Video in einem Fenster abgespielt werden, so wird die Grafikausgabe des umgebenden Desktops und aller Fenster weiter auf der entfernten Maschine erzeugt und als Bitmap an den Client übertragen. Das Video selbst wird dagegen zunächst direkt an den Client gestreamt, dort mit dem lokalen Codec dekodiert und nahtlos in die Bildschirmausgabe des Remote-Desktop integriert. Diese Technik namens HDX MediaStream unterstützt zahlreiche Audio- und Video-Formate, darunter AVI, MP4, DivX, OGG, WMV, WAV, MP3, AAC, WMA und auch Flash." (vgl. [Knermann, 2010a]).

Auch beim Server-seitigen Rendern grafischer Inhalte hat das ICA Protokoll Verbesserungen erfahren, insbesondere mit einer als „HDX 3D Pro" bezeichneten Erweiterung, die es ermöglicht, im Server vorhandene Grafikchipsätze (Graphics Processing Unit (GPU)) für die

Kompression der zu übertragenden Bildschirminhalte einzusetzen[18]. Eine weitere im Jahr 2011 eingeführte Erweiterung namens „Multi-Stream ICA" sieht die Möglichkeit vor, den Datenstrom des ICA Protokolls auf mehrere TCP/IP-Verbindungen aufzuteilen[19]. Dies dient dem Zweck, auf Netzwerkebene Qualitiy of Service (QoS) Richtlinien auf die einzelnen Verbindungen anwenden zu können und so beispielsweise die Übertragung von Echtzeit-Daten, wie Audio und Video, höher zu priorisieren oder die virtuellen Kanäle für den Transfer von Dateien und Druckaufträgen niedriger zu priorisieren als die Übertragung der übrigen Grafikausgabe.

Unabhängig davon, ob Terminal Server Sitzungen via ICA oder RDP betrieben werden, lassen sich Windows Serversysteme mit den zuvor genannten Technologien nach dem Betriebsmodell der entfernten Präsentation im Multi-User-Betrieb nutzen und so effizient auslasten. Im Sinne weiterhin optimierter Auslastung vorhandener Hardware wurden in jüngerer Vergangenheit verstärkt Lösungen entwickelt und implementiert, um Serversysteme allgemein und insbesondere auch Terminal Server von der unterliegenden Hardware zu abstrahieren und zu virtualisieren.

2.2 (Server-)Virtualisierung

Allgemein formuliert hat Virtualisierung zum Ziel, scheinbar vorhandene Netzwerke, Systeme, Geräte, Hardware-Komponenten, Verbindungen und Zustände zu simulieren. Dies dient unter anderem dem Zweck, vorhandene Ressourcen zur effizienteren Nutzung anders zu segmentieren und darzustellen, als dies den tatsächlich vorhandenen physischen Gegebenheiten entspricht.

2.2.1 Historischer Hintergrund

Wie auch die Konzepte und Techniken der entfernten Präsentation hat die Virtualisierung ihren Ursprung in der Welt der Großrechner. So wurde zwischen 1964 und 1966 von IBM ein Betriebssystem für virtuelle Maschinen namens CP/CMS auf dem Mainframe IBM System/360 Model 40 implementiert, dessen Wurzeln sogar bis auf Entwicklungen in den 1950er Jahren zurückgeht.

Dieses Betriebssystem bestand aus dem Conversational Monitor System (CMS), einem Single-User-System, und dem Kontrollprogramm CP-40, das es der Hardware ermöglichte, mehrere Instanzen ihrerselbst als virtuelle Maschinen (VM) zu simulieren, in denen jeweils eine Installation von CMS betrieben werden konnte: „CP is an operating system that uses a computing machine to simulate multiple copies of the machine itself. These copies, or virtual machines, are each controlled like the real machine by their own operating systems. CMS is an operating system that supports the interactive use of a computing machine by one person." (vgl. [Creasy, 1981], S. 483). Dieses Betriebssystem wurde nur auf einer einzigen Hardware im

[18] http://blogs.citrix.com/2011/09/06/hdx-3d-pro-takes-another-leap-forward/ (abgerufen am 01.02.2016).
[19] http://blogs.citrix.com/2011/08/25/enhanced-qos-via-multi-stream-ica/ (abgerufen am 01.02.2016).

IBM Cambridge Scientific Center installiert. Das nachfolgende Betriebssystem Virtual Machine Facility/370, kurz VM/370, bestehend aus dem Kontrollprogramm CP-67 und CMS, erschien 1972 für das IBM System/370 und erfuhr eine deutliche stärkere Verbreitung. VM/370 war bereits in der Lage, mehrere Benutzer nach dem Funktionsprinzip des sogenannten Time-Sharings zu bedienen[20] und virtuellen Hauptspeicher zu nutzen, d. h. Teile des Hauptspeichers auf günstigeren Massenspeicher auszulagern. Damit nahmen System/370 und VM/370 die Architektur betreffend die Merkmale vieler späterer Computersysteme und Infrastrukturen vorweg[21]. Nur wenige weitere Systeme waren zu dieser Zeit in der Lage, virtuelle Maschinen zu betreiben. Die Mehrheit der damals erhältlichen Computersysteme vermochte dies nicht, obwohl Goldberg in seinem theoretischen Modell für eine Architektur von virtuellen Maschinen bereits zahlreiche auch heute noch bestehende Vorteile von Virtualisierung anführte (vgl. [Goldberg, 1973], S. 75 f.), als da wären:

- Verbessern und Testen von Betriebssystem-Software bei gleichzeitigem Produktiv-Betrieb des Systems
- Ausführen verschiedener Betriebssysteme oder Versionen eines Betriebssystems
- Ausführen einer virtuellen Konfiguration, die vom realen System abweicht, z. B. mehr Hauptspeicher oder Prozessoren als tatsächlich vorhanden
- Ändern der Hardware, ohne vorhandene Betriebssysteme anpassen zu müssen
- Erhöhung der Zuverlässigkeit und Sicherheit von Anwendungen, die dies erfordern

Gemeinsam mit Popek beschrieb Goldberg daraufhin formelle Anforderungen an eine Systemarchitektur der dritten Generation[22], die erfüllt sein müssen, damit diese Architektur in der Lage ist, virtuelle Maschinen zu betreiben (vgl. [Popek, 1974]). Die Überlegungen lassen sich auch auf Architekturen der vierten Generation übertragen.

2.2.2 Virtual Machine Monitor

Den Kern bildet das Konzept eines sogenannten Virtual Machine Monitors (VMM)[23], der als Abstraktionsschicht zwischen der Hardware und einer oder mehreren virtuellen Maschinen vermittelt (Abbildung 2-9). Heute ist der Begriff Hypervisor als Synonym für den VMM verbreiteter.

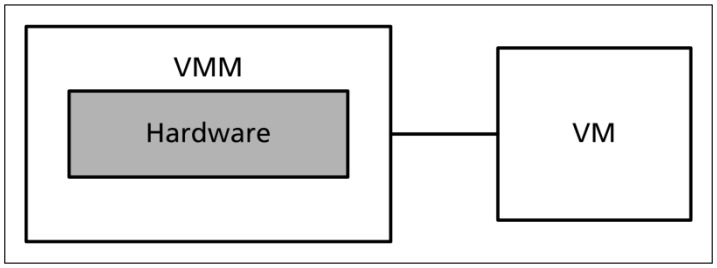
[20] Die Pionierarbeit für ein Mehrzweck-Time-Sharing-System wurde Anfang der 1960er Jahre am MIT erbracht (vgl. [Creasy, 1981], S. 484). Das dort entwickelte „Compatible Time-sharing System" (CTSS) ermöglichte es als eines der ersten seiner Art mehreren Benutzern, ein Computersystem zeitgleich zu nutzen, indem der vorhandene Prozessor den Benutzern in schneller Folge abwechselnd zugeteilt wurde.

[21] „It is precisely because these features also characterize VM/370 to its users that this system design should be viewed in the modern context. VM/370, with its collection of interconnected, independent machines each serving one user, can provide architectural compatibility with the future's networks of personal computers." (vgl. [Creasy, 1981], S. 484).

[22] Die erste Generation von Computern bildeten auf Röhren basierende Systeme, während in der zweiten Generation diese Röhren durch Transistorschaltungen ersetzt wurden. Computersysteme der dritten Generation umfassen die auf integrierten Schaltkreisen, also auf Siliziumchips bzw. Halbleiter reduzierten Transistorschaltungen, basierenden Systeme. Die vierte Generation bezeichnet die bis heute üblichen Systeme auf Basis von Mikroprozessoren, während der Begriff der fünften Generation für Entwicklungen im Bereich der Künstlichen Intelligenz (KI) verwendet wird.

[23] Die Firma Microsoft verwendet ebenfalls das Akronym VMM. Dieses steht im Kontext der Microsoft Virtualisierungsprodukte aber für den „Virtual Machine Manager", eine Lösung für das Management von virtuellen Maschinen (vgl. [Tullock, 2010], S. 313 ff.).

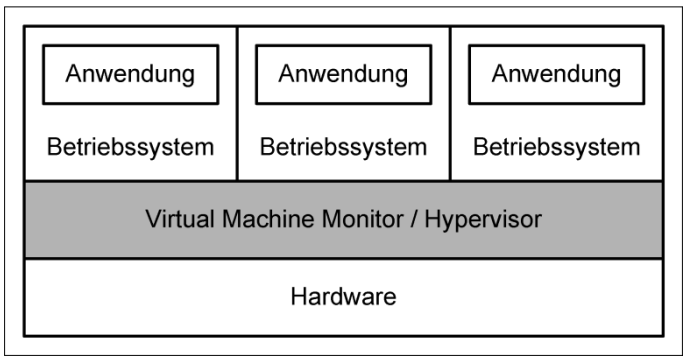
Abbildung 2-9: Virtual Machine Monitor (VMM)



Quelle: Eigene Darstellung, nach [Popek, 1974], S. 413

Ein solcher Virtual Machine Monitor oder Hypervisor muss drei Anforderungen erfüllen (vgl. [Popek, 1974], S. 413). So soll erstens eine virtuelle Maschine (VM) eine identische Kopie der tatsächlichen Hardware darstellen, in dem Sinne, dass sich ein Betriebssystem innerhalb einer VM genauso verhält, als würde es direkt auf der unterliegenden Hardware ausgeführt. Die zweite Anforderung verlangt Effizienz, also dass ein System innerhalb einer VM annähernd so performant ausführbar ist wie auf der physischen Hardware, womit Emulation (vgl. Kapitel 2.2.3) ausscheidet. Um dieses Ziel zu erreichen, ist gefordert, dass eine signifikante Untermenge des Befehlssatzes des virtuellen Prozessors direkt und ohne Eingreifen des VMM vom physischen Prozessor der Hardware ausgeführt wird. Demgegenüber verlangt die dritte Anforderung der Isolierung, dass der VMM die übrige Hardware, wie Hauptspeicher und Peripherie, verwaltet, damit eine VM nicht auf Ressourcen zugreifen kann, die ihr nicht explizit zugeteilt oder anderweitig alloziert sind.

Abbildung 2-10: Typ 1 Virtual Machine Monitor

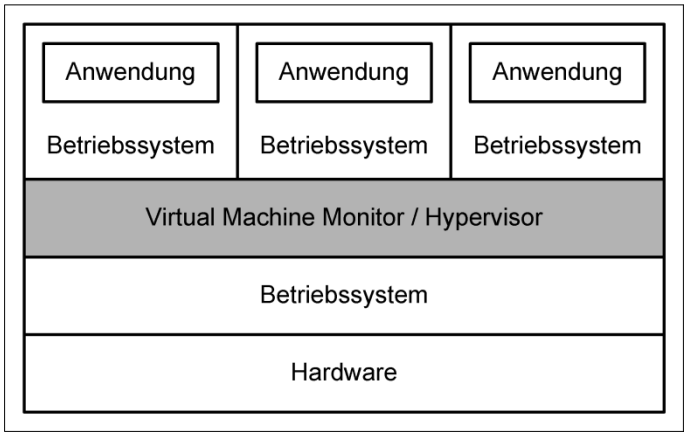


Quelle: Eigene Darstellung, nach [IBM, 2005]

Ein klassischer VMM verhindert dies, indem er virtuelle Maschinen mit reduzierten Rechten ausführt. Immer wenn eine VM versucht, privilegierte Instruktionen auszuführen, fängt der VMM die entsprechenden Instruktionen ab und setzt diese in sicherer Weise von der virtuellen auf die physische Maschine um. Dieses als „Trap-and-Emulate“ bezeichnete Verfahren war in den Anfängen der Virtualisierungstechniken derart verbreitet, dass es lange Zeit als die einzig praktikable Methode für Virtualisierung überhaupt angesehen wurde (vgl. [Adams, 2006]).

Es werden weiterhin zwei Arten von VMM unterschieden (vgl. [Goldberg, 1972]). So wird ein VMM vom Typ 1 direkt auf einer physischen Hardware ausgeführt und hat entsprechend vollständigen Zugriff darauf (Abbildung 2-10). In diese Kategorie fällt das bereits erwähnte Kontrollprogramm von VM/370 als historischer Vorgänger heutiger Virtualisierungslösungen. Ein VMM oder Hypervisor vom Typ 2 dagegen benötigt zunächst ein herkömmliches Betriebssystem als Basis – das Hostsystem, welches Peripheriegeräte und Hauptspeicher verwaltet (Abbildung 2-11).

Abbildung 2-11: Typ 2 Virtual Machine Monitor



Quelle: Eigene Darstellung, nach [IBM, 2005]

Typ 2 Hypervisoren kommen hauptsächlich auf Clientsystemen zum Einsatz, während Typ 1 Hypervisoren auf Grund ihrer gegenüber dem Typ 2 höheren Performanz, Verfügbarkeit und Sicherheit für die Virtualisierung von Serversystemen in Rechenzentren eingesetzt werden. Entsprechend werden im weiteren Verlauf nur Hypervisoren vom Typ 1 Gegenstand der Betrachtungen sein. In Abgrenzung dazu werden im Folgenden zunächst weitere Virtualisierungstechniken diskutiert, welche nicht der Arbeitsweise eines klassischen VMM entsprechen. Da die x86 Architektur vor dem Hintergrund ihrer historischen Entwicklung bis hin zum Intel Pentium III Prozessor nicht in der Lage war, Hardwareunterstützung für den sicheren Betrieb eines VMM zu bieten (vgl. [Robin, 2000]), waren diese Virtualisierungstechniken insbesondere für x86 Plattformen relevant und kommen auch heute noch in bestimmten Anwendungsfällen zum Einsatz.

2.2.3 Emulation

Eine Virtualisierungslösung wird als Emulation bezeichnet, wenn innerhalb einer VM die Hardware vollständig in Software abgebildet wird. Eine solche Lösung wird auch als Complete Software Interpreter Machine (CSIM) bezeichnet, da sämtliche Prozessorinstruktionen von der Software ausgeführt und nicht direkt an den physischen Prozessor übergeben werden. Das Betriebssystem, welches innerhalb der VM installiert wird, kann in diesem Fall durchaus für

einen Prozessortyp entwickelt sein, der sich vom tatsächlich vorhandenen physischen Prozessor grundlegend unterscheidet[24]. Mit der völligen Abstraktion von der unterliegenden Plattform wird zwar weitestgehend Hardware-Unabhängigkeit erreicht, da es in diesem Fall aber unmöglich ist, Instruktionen direkt auf dem physischen Prozessor auszuführen, kann ein Emulator die ersten beiden Anforderungen von Popek und Goldberg (vgl. Kapitel 2.2.1) nicht erfüllen. In der Praxis bedeutet dies insbesondere, dass eine auf diese Weise virtualisierte Maschine gegenüber ihrem physischen Pendant wesentlich geringere Performanz aufweist. „Im professionellen Umfeld sind Emulatoren zu langsam, um kritische Applikationen zu steuern, da eine Umsetzung von Prozessorbefehlen (etwa von Little Endian auf Big Endian und vice versa[25]) und Speicherregistern enorme Ressourcen verschlingt." (vgl. [Zimmer, 2006], S. 64) Emulatoren werden entsprechend im Folgenden nicht weiter betrachtet.

2.2.4 Betriebssystem-Virtualisierung (Container)

Die Betriebssystem-Virtualisierung oder auch Virtualisierung auf Betriebssystemebene – nach der englischen Bezeichnung „Operating System-Level Virtualization" – hat zum Ziel, ein und dasselbe Betriebssystem in mehrere voneinander abgeschottete Betriebsumgebungen zu partitionieren. Im Unterschied zum klassischen VMM ist die Virtualisierungsschicht nicht zwischen Hardware und Betriebssystem, sondern zwischen Betriebssystem und den Anwendungen angesiedelt (vgl. (Yu, 2006]).

Dieser Ansatz ist weniger weitreichend als ein klassischer VMM, da innerhalb der virtuellen Umgebung kein vollständiges Betriebssystem, insbesondere kein von der unterliegenden Basis abweichendes Betriebssystem, installiert werden kann. Auf der anderen Seite geht die Betriebssystem-Virtualisierung aber über die Möglichkeiten der Anwendungsvirtualisierung (vgl. Kapitel 2.3.3) deutlich hinaus, da nicht nur Dateien und – spezifisch für die Microsoft Windows Plattform – Registrierungseinträge, sondern auch Systembibliotheken und Teile der Interprozesskommunikation (IPC) virtualisiert werden können.

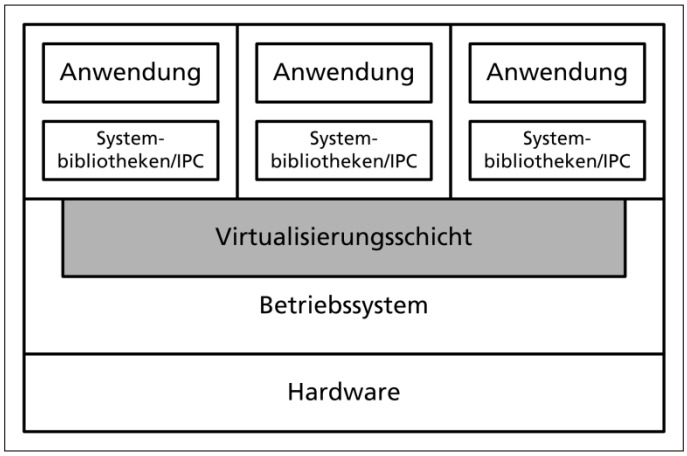
Auf Unix-artigen Betriebssystemen ist die Betriebssystem-Virtualisierung als etablierte Technik anzusehen, im einfachsten Fall als „Change root" (chroot) Umgebung. Weitere verbreitete Ausprägungen dieses Ansatzes sind beispielsweise die sogenannten Jails unter FreeBSD, Sun Solaris Zones sowie die Linux-Projekte VServer, OpenVZ und das Linux Containers (LXC) Framework (vgl. [Plura, 2014]). Eine weitere, relativ junge Entwicklung aus dem Linux-Umfeld

[24] Praktische Beispiele für Emulatoren sind das „Bochs IA-32 Emulator Project" (http://bochs.sourceforge.net (abgerufen am 01.02.2016)) oder der „QEMU Open Source Processor Emulator" (http://qemu.org (abgerufen am 01.02.2016)), ebenso wie verschiedene weitere Projekte, die es sich etwa zum Ziel gesetzt haben, ältere Hardwareplattformen, wie den Commodore C64, Amiga oder Motorola 68k und PowerPC basierende Apple Macintosh Systeme auf x86 Plattformen zu emulieren.

[25] Die Begriffe „Little Endian" und „Big Endian" bezeichnen die Byte-Reihenfolge, nach der Zahlenwerte im Speicher abgelegt werden. Eine solche Reihenfolge ist festzulegen, sobald ein zu speichernder Wert das Fassungsvermögen der kleinsten adressierbaren Speichereinheit, i. d. R. ein Byte, übersteigt. Systeme, welche die Reihenfolge „Little Endian" verwenden, speichern das Bit mit der niedrigsten Wertigkeit zuerst, so z. B. sämtliche Systeme, die die x86 Architektur verwenden. Die „Big Endian" Reihenfolge beginnt dagegen mit dem Bit mit der höchsten Wertigkeit und findet beispielsweise bei Motorola 68k Systemen Anwendung. Soll nun etwa ein älteres Apple Mac OS als Motorola 68k System auf einer x86 Plattform emuliert werden, muss der Emulator von der einen auf die andere Speicheradressierung übersetzen, was zwangsläufig zu Leistungseinbußen führt.

ist die Virtualisierungslösung Docker[26], die nicht auf einer chroot-Umgebung basiert, sondern die zu isolierende Anwendung abgesehen vom Kernel mit allen nötigen Komponenten in ein Image verpackt (vgl. [Leemhuis, 2014]). Allen diesen Lösungen ist also gemein, dass innerhalb der virtuellen Umgebung kein vollständiges Betriebssystem gestartet wird, sondern sich sämtliche Container den Kernel des unterliegenden Betriebssystems teilen. Der verbreitetste Anwendungsfall der Betriebssystem-Virtualisierung findet sich im Betrieb von Webhosting-Plattformen.

Abbildung 2-12: Betriebssystem-Virtualisierung



Quelle: Eigene Darstellung

Für die Microsoft Windows Plattform ist mit der Feather-Weight Virtual Machine (FVM) eine Implementierung (vgl. [Yu, 2006] und [Yu, 2007]) mit einigen Anwendungsfällen (vgl. [Yu, 2008]) beschrieben. Das entsprechende Open Source Projekt fvm-rni"[27] wurde bislang nur für die Microsoft Betriebssysteme Windows 2000 und Windows XP implementiert und befindet sich derzeit im Prototypenstadium. Mit den Parallels Virtuozzo Containers (PVC) existiert eine weitere, produktiv einsetzbare Implementierung für die Windows Plattform, die sich von der FVM hauptsächlich durch die Art des Zugriffs auf gemeinsam genutzte Ressourcen unterscheidet (vgl. [Yu, 2007], S. 11).

Ein Nachteil des Container-Ansatzes unter den Gesichtspunkten der Sicherheit und Isolierung der Anwendungen und Sitzungen voneinander ist, dass die Basis *ein* Kernel mit *einem* monolithischen Adressraum bildet, über dem lediglich eine Abstraktionsschicht mit zusätzlichen Sicherheitsmaßnahmen etabliert wird (vgl. [Porter, 2011], S. 302). Mit dem auf Basis von Windows 7 entwickelten Projekt Drawbridge[28] hat Microsoft Research einen weitergehenden Ansatz vorgestellt, der die Architektur des unterliegenden Betriebssystem verändert, so dass jeder Applikation eine eigene Umgebung mit eigenem Adressraum zur

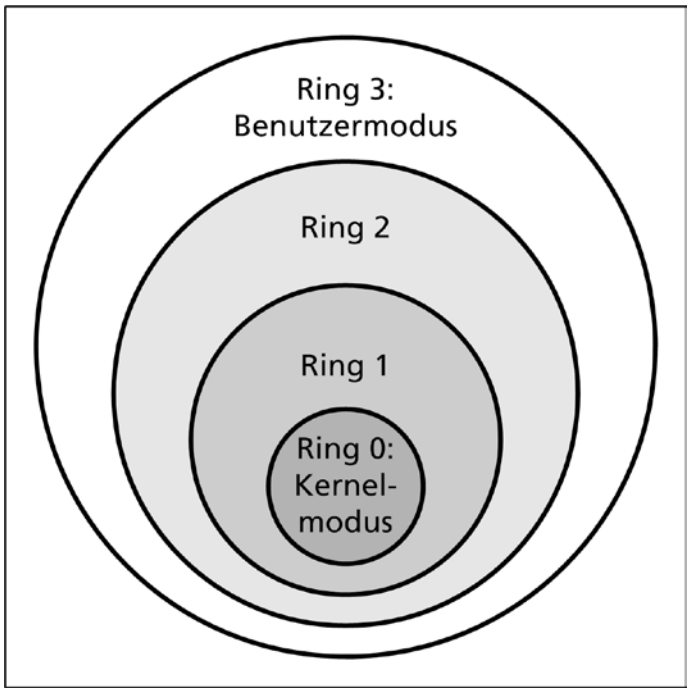
[26] https://www.docker.com (abgerufen am 01.02.2016).
[27] http://fvm-rni.sourceforge.net (abgerufen am 01.02.2016).
[28] http://research.microsoft.com/apps/pubs/default.aspx?id=141071 (abgerufen am 01.02.2016).

Verfügung steht. Dieses Projekt befindet sich allerdings explizit in einem experimentellen Stadium, und es ist nach derzeitigem Stand der Technik noch nicht abzusehen, wann und ob entsprechende Technologien in produktiv einsetzbare Betriebssysteme Einzug halten werden.

2.2.5 Binary Translation

Wie in Kapitel 2.2.2 bereits erwähnt, sind ältere x86 Prozessoren nicht in der Lage, Anforderungen für den Betrieb eines klassischen VMM zu erfüllen. Vier Eigenschaften wurden hierzu beschrieben (vgl. [Goldberg, 1972]). So muss ein Prozessor über zwei Betriebsmodi verfügen, einen privilegierten Kernelmodus sowie einen nicht-privilegierten Benutzermodus. Es muss für nicht-privilegierte Programme auf sichere Weise möglich sein, privilegierte Systemfunktionen aufzurufen, und es müssen Schutzmechanismen oder Methoden zur Verwaltung virtuellen Speichers, wie Segmentierung oder Paging, vorhanden sein. Des Weiteren muss das Eingabe-/Ausgabe-Subsystem mittels Interrupts in der Lage sein, asynchron mit dem Prozessor zu kommunizieren.

Abbildung 2-13: Prozessor-Ringe der x86 Architektur



Quelle: Eigene Darstellung

Dazu wurde im Kontext der x86 Architektur der geschützte Modus des Prozessors mit vier Stufen absteigender Priorität, den sogenannten Ringen 0-3, implementiert (Abbildung 2-13). Ein Betriebssystem läuft im Regelfall mit der höchsten Priorität im Kernelmodus auf Ring 0 und hat entsprechend uneingeschränkten Zugriff auf die Hardware, während die Anwendungen im Benutzermodus auf Ring 3 laufen (vgl. [Ahnert, 2007], S. 42). Ring 1 und 2 bleiben typischerweise, d. h. zumindest unter Microsoft Windows und Linux-Derivaten, ungenutzt.

Wird ein Betriebssystem nun innerhalb einer VM installiert, so läuft diese auf einem Ring mit niedrigerer Priorität, in der Regel ebenfalls im Ring 3 und hat somit keinen direkten Zugang zur unterliegenden Hardware. Ein VMM, der auf einer solchen Architektur betrieben wird, muss entsprechend privilegierte Systemfunktionen – sensible Aktionen, welche direkt auf Prozessorregister oder Speicheradressen zugreifen – innerhalb einer virtuellen Maschine abfangen („Trap") und in sicherer Weise an den physischen Prozessor übergeben („Emulate"). Da im Befehlssatz des Pentium-Prozessors allerdings auch nicht-privilegierte Funktionen, d. h. Funktionen, welche von einem klassischen VMM nicht erkannt und behandelt werden, sensible Aktionen ausführen (vgl. [Robin, 2000] und [Ahnert, 2007], S. 43), müssen diese in Software abgebildet werden. Der Ansatz der Binary Translation (BT), wie sie beispielsweise in den Hypervisoren von VMware zum Einsatz kommt, ergänzt den VMM daher um einen Interpreter, so dass auch die x86 Architektur in die Lage versetzt wird, auf sichere Weise virtuelle Maschinen zu betreiben (vgl. [Adams, 2006], S. 3): „A VMM built around a suitable binary translator can virtualize the x86 architecture and it is a VMM according to Popek and Goldberg."

Der BT Interpreter liest den kompilierten x86 Binärcode zur Laufzeit aus dem Hauptspeicher der virtuellen Maschinen und überwacht diesen in Hinsicht auf kritische Aufrufe. Während unkritische Funktionen identisch umgesetzt werden, werden die problematischen Funktionen vom Interpreter durch unkritische ersetzt. Da die Mehrzahl der Funktionen im Benutzermodus unkritisch ist, empfiehlt sich eine Kombination von BT und direkter Ausführung, um Leistungsverluste zu minimieren (vgl. [Adams, 2006], S. 4): „…we observe that BT is not required for safe execution of most user code on most guest operating systems. By switching guest execution between BT mode and direct execution as the guest switches between kernel- and user-mode, we can limit BT overheads to kernel code and permit application code to run at native speed."

2.2.6 Paravirtualisierung

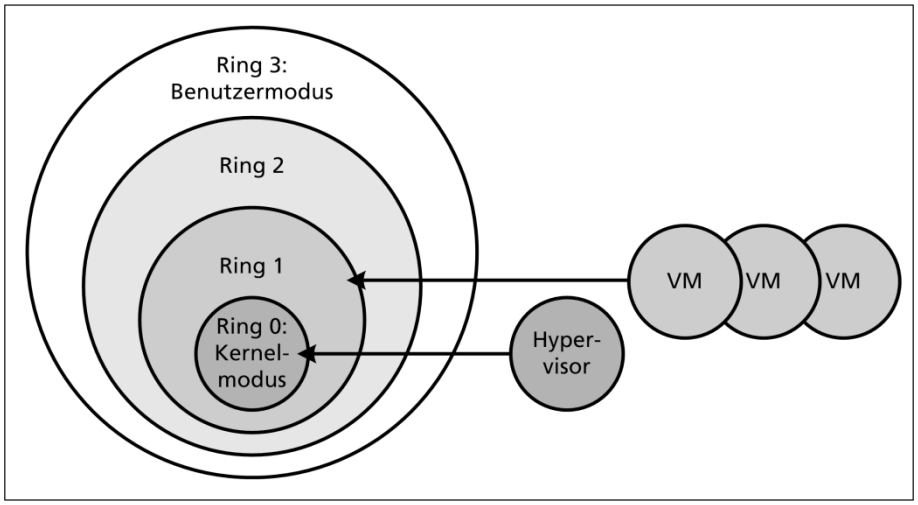
Auch das Konzept der Paravirtualisierung hat seine Wurzeln in der Historie und fand unter anderem bereits in IBMs VM/370 Verwendung (vgl. [Goldberg, 1974]), wurde dort aber nur in geringem Umfang eingesetzt, um das Ziel, ältere Betriebssysteme zu virtualisieren, nicht zu verfehlen (vgl. [Whitaker, 2002], Abs. 6.3). Eine Paravirtualisierung arbeitet ähnlich einem klassischen VMM, allerdings wird die Anforderung, dass eine VM identisch zur physischen Hardware sein soll, explizit nicht erfüllt. Die virtuelle Hardware der VM ist der physischen Basis lediglich ähnlich, was bezogen auf die x86 Architektur bedeutet, dass Betriebssysteme für den Einsatz innerhalb einer VM derart modifiziert werden müssen, dass sie keine kritischen Instruktionen mehr im Ring 0 des Prozessors ausführen.

Dieses Prinzip wurde für die x86 Architektur insbesondere durch den Denali Isolation Kernel geprägt (vgl. [Whitaker, 2002/a/b]), die prototypische Implementierung einer Betriebsumgebung für virtuelle Maschinen. Sämtliche kritischen Funktionen werden innerhalb der VM durch unkritische ersetzt, „…virtual instructions that have no counterpart in the physical architecture; these are conceptually similar to OS system calls, except that they are

non-blocking and they operate at the architectural level instead of at the level of OS abstractions“ (vgl. [Whitaker, 2002], Abs. 3.1).

Größere Verbreitung hat die Paravirtualisierung schließlich durch das Open Source Projekt Xen[29] erfahren, aus dem beispielsweise die Implementierung des Citrix XenServer[30] hervorgegangen ist. Der Xen-Hypervisor läuft im Prozessor-Ring 0. Virtuelle Maschinen werden im ansonsten ungenutzten Ring 1 betrieben (Abbildung 2-14).

Abbildung 2-14: Paravirtualisierung



Quelle: Eigene Darstellung

Der Kernel eines Gastbetriebssystems, das innerhalb einer VM installiert werden soll, muss dazu bereits während dessen Kompilierung so modifiziert werden, dass keine kritischen Instruktionen im Kernel-Modus mehr verwendet werden. Dies betrifft neben dem Prozessor selbst sowohl die Hauptspeicherverwaltung als auch sämtliche Zugriffe auf Eingabe-/ Ausgabe-Geräte (vgl. [Barham, 2003], Abs. 2.1).

Als Vorteil der Paravirtualisierung im Xen-Hypervisor lässt sich neben deutlichen Geschwindigkeitsvorteilen gegenüber einer Emulation der virtuellen Hardware verbuchen, dass zwar die Schnittstelle zwischen VMM und virtualisiertem Betriebssystem angepasst werden muss, die Schnittstelle zwischen Betriebssystem und den Anwendungen, die im Ring 3 des Prozessors ausgeführt werden, aber unverändert bestehen bleibt. Standardsoftware kann somit ohne weitere Anpassungen auf virtuelle Systeme übernommen werden: „…we do not require changes to the application binary interface (ABI), and hence no modifications are required to guest applications. (…) Xen is intended to scale to approximately 100 virtual machines running industry standard applications and services.“ (vgl. [Barham, 2003], S. 2).

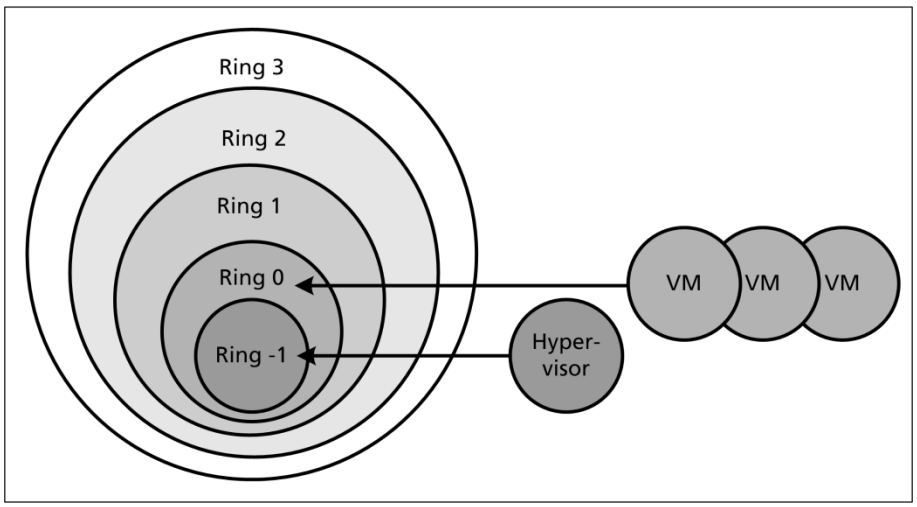
[29] The Xen hypervisor: http://www.xenproject.org (abgerufen am 01.02.2016).
[30] Der Citrix XenServer (http://www.citrix.de/products/xenserver/overview.html (abgerufen am 01.02.2016)) hat seinen Ursprung in der von den Xen-Entwicklern gegründeten Firma XenSource, welche im Jahr 2007 von Citrix Systems übernommen wurde.

Doch mit der notwendigen Anpassung der Gastbetriebssysteme liegt auch der größte Nachteil auf der Hand, dass nämlich Betriebssysteme, deren Quelltext nicht in veränderbarer Form vorliegt, wie es insbesondere für die Microsoft Windows Familie der Fall ist, durch Dritte nicht vollständig paravirtualisiert werden können. Aus diesem Grund setzen Xen, wie auch der Microsoft eigene Hypervisor Hyper-V[31] heute neben Paravirtualisierung verstärkt auf die Hardware-gestützte Virtualisierung.

2.2.7 Hardware-gestützte Virtualisierung

Die hardware-gestützte Virtualisierung, engl. „hardware-assisted virtualization", hat zum Ziel, die x86 Architektur derart zu verändern, dass darauf Virtualisierung im Sinne eines klassischen VMM möglich wird. Dazu haben die Prozessor-Hersteller Intel und AMD unabhängig voneinander ihre Prozessoren um zusätzliche Funktionen erweitert (vgl. [Adams, 2006], Abs. 4). Bei Intel geschah dies unter dem Projektnamen „Vanderpool", bei AMD unter dem Label „Pacifica", in beiden Fällen mit dem Ziel, die Ausführung unmodifizierter Betriebssysteme innerhalb einer virtuellen Maschine zu erlauben.

Abbildung 2-15: Hardware-gestützte Virtualisierung mit Intel VT-x



Quelle: Eigene Darstellung

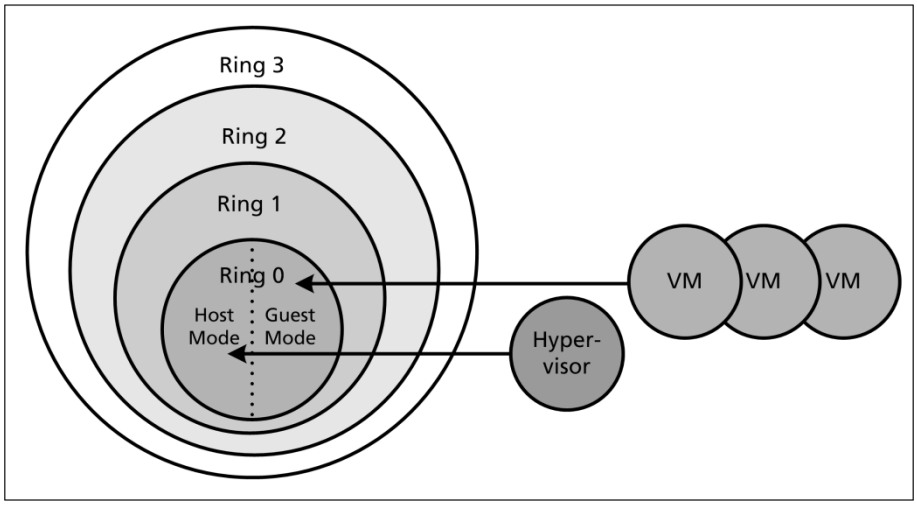
Aus Intels Projekt Vanderpool ist die „Intel Virtualization Technology (IVT)" hervorgegangen, welche erstmals im Jahr 2005 verbaut und ausgeliefert wurde, konkret als sogenannte „VT-x" Erweiterung für die x86 Architektur (Intel 64 bzw. IA-32[32]) sowie als „VT-i" Erweiterung für

[31] http://www.microsoft.com/germany/windowsserver2008/2008/technologien/hyper-v.mspx (abgerufen am 01.02.2016).
[32] Die Architekturen Intel 64 und IA-32 bezeichnen Intels Prozessoren für die x86 Plattform. Details hierzu finden sich in [Intel, 2011].

den Itanium Prozessor (IA-64[33]). Die IVT erfordert neben einem passenden Prozessor auch die Unterstützung durch Chipsatz und BIOS des jeweiligen Mainboards sowie natürlich einen daran angepassten Hypervisor. VT-x führt einen zusätzlichen virtuellen Prozessor Ring unterhalb des regulären Ring 0 ein, in dem der VMM betrieben wird (Abbildung 2-15), so dass die VM unmodifiziert im Ring 0 laufen können und konkurrierende Zugriffe bzw. kritische Funktionsaufrufe in sicherer Weise vom unterliegenden VMM behandelt werden können.

Als zusätzliche Ergänzung kam in jüngerer Vergangenheit auf Ebene des Chipsatzes bzw. Mainboards die „Intel Virtualization Technology For Directed I/O" (VT-d) hinzu, welche einen darauf ausgelegten VMM in die Lage versetzt, Zugriffe von VM auf den Hauptspeicher, Eingabe-/Ausgabe-Geräte sowie Interrupts und Direct Memory Access Aufrufe (DMA) in sicherer und isolierter Weise direkt zu unterstützen (vgl. [Intel, 2011a], Abs. 2.5). Einen weiteren Baustein bildet die „Intel Virtualization Technology For Connectivity" (VT-c), welche der Optimierung der Netzwerk-Eingabe und -Ausgabe dient, indem der Hypervisor von der Verarbeitung von Netzwerkoperationen in Software entlastet wird und diese Aufgaben stattdessen im Chipsatz, d. h. direkt in der Hardware, implementiert werden (vgl. [Gai, 2010], S. 65 ff.).

Abbildung 2-16: Hardware-gestützte Virtualisierung mit AMD-V



Quelle: Eigene Darstellung

Die aus AMDs Projekt Pacifica entstandene Technologie „AMD Virtualization (AMD-V)" als Erweiterung der AMD64 Architektur kam erstmalig in 2006 zum Einsatz und ähnelt dem Ansatz von Intel VT-x – zumindest im Hinblick auf das Ziel, unmodifizierte Gastbetriebssysteme in den VM einzusetzen. Um dieses Ziel zu erreichen, erweitert AMD-V die Funktionalität des

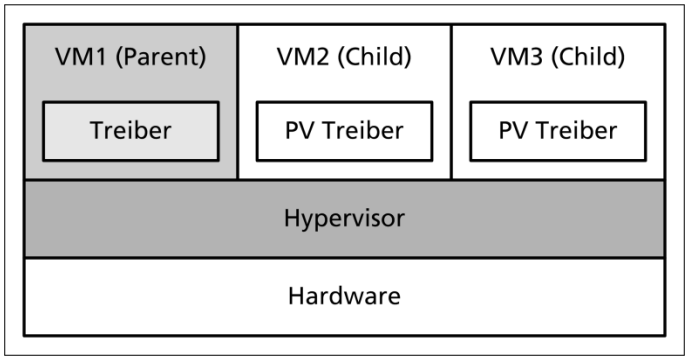
[33] Die IA-64 Architektur bezieht sich auf den zur x86 Architektur weitestgehend inkompatiblen Itanium Prozessor. Details hierzu finden sich in [Intel, 2010]. Da der Itanium-Prozessor seit 2011 kaum mehr gebräuchlich ist und insbesondere von aktuellen Versionen der Microsoft Windows Betriebssysteme nicht mehr unterstützt wird, bleibt er im Folgenden unberücksichtigt und es steht die x86 Architektur im Fokus der Betrachtungen.

Prozessor-Rings 0 (Abbildung 2-16). Neben den regulären Modus, in dem der VMM operiert, bei AMD als „Host-Mode“ bezeichnet, tritt der sogenannte „Guest-Mode“ für die virtuellen Maschinen. In diesem Modus können die VM gefahrlos kritische Instruktionen aufrufen, welche anschließend in sicherer Weise, und ohne andere VM zu beeinträchtigen, ausgeführt werden können (vgl. [Zeichick, 2006] und [Adams, 2006], Abs. 4.1). Entsprechend Intel VT-d verfügt auch die AMD Architektur mit der „AMD IO Virtualization Technology (IOMMU)“ über eine Erweiterung für das Speichermanagement, die Behandlung von DMA Transaktionen sowie allgemein direkte Zugriffe auf Eingabe-/Ausgabe-Geräte (vgl. [AMD, 2011]).

Heutige VMM, wie der bereits erwähnte Xen Hypervisor oder die Typ 1 Hypervisoren von VMware, nutzen die Erweiterungen von Intel respektive AMD, um unmodifizierte Gastsysteme zu betreiben. So wird es beispielsweise ohne den Bedarf für Eingriffe in den Kernel möglich, auch die Systeme der Microsoft Windows Familie zu virtualisieren. Dabei kommt die hardware-gestützte Virtualisierung, je nach Produkt, fallweise in Kombination mit BT oder Paravirtualisierung zum Einsatz.

So ergänzen der Xen Hypervisor und ebenso Microsoft Hyper-V einen per hardware-gestützter Virtualisierung ausgeführten Windows-Kernel in einer VM, z. B. mit paravirtualisierten Treibern für Grafik- und Netzwerkkarte, um die Leistung zu steigern und zusätzliche Funktionen zu realisieren. Der virtuellen Maschine gegenüber erscheinen die entsprechenden Geräte als generische Grafikkarte, Netzwerkkarte oder anderweitige Hardware, so dass die Virtualisierungsschicht von der tatsächlich vorhandenen, physischen Hardware abstrahiert und die paravirtualisierten Treiber innerhalb der VM somit nicht an die physischen Gegebenheiten angepasst werden müssen.

Abbildung 2-17: Microkernel Hypervisor

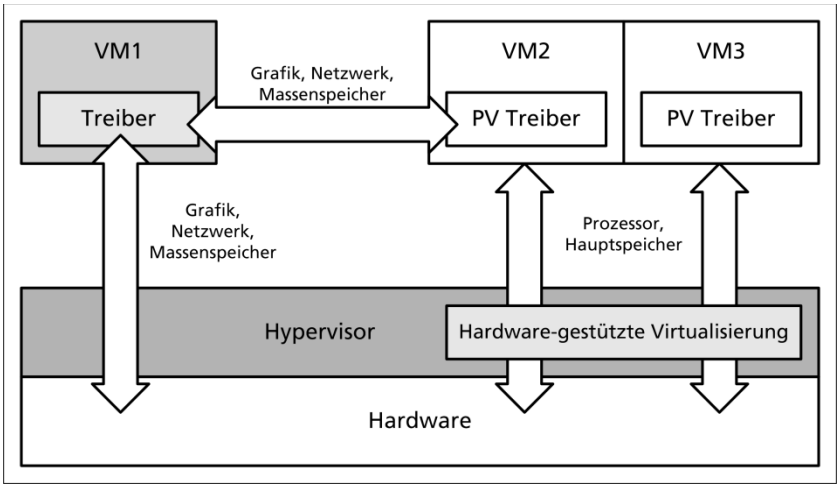


Quelle: Eigene Darstellung, nach [Tullock, 2010], S. 26

Hyper-V und Xen setzen dabei auf eine Microkernel-Architektur auf. Dabei werden die eigentlichen Treiber für den Zugriff auf die physischen Geräte nicht im Hypervisor selbst implementiert, sondern in einer privilegierten Kontroll-VM (Abbildung 2-17). Diese VM wird im Kontext von Hyper-V als Parent Partition bezeichnet, im Zusammenhang mit Xen auch als Control Domain, Domain-0 oder kurz Dom0. Vorteile dieses Ansatzes sind, dass der Hypervisor selbst mit möglichst geringem Overhead auskommt und zudem die Angriffsfläche des

Hypervisors minimiert wird, da kein Code von Drittanbietern – z. B. Hersteller-Treiber für diverse Hardware – direkt in den Hypervisor geladen werden muss (vgl. [Tullock, 2010], S. 26 ff.). Andererseits bedeutet dies, dass die Kontroll-VM über die entsprechenden Treiber verfügen muss und die übrigen VM – beim Hyper-V als Child Partition, im Fall von Xen als DomU bezeichnet, wobei das „U" für „unprivileged" bzw. den User-Modus des Prozessors steht – nur mittelbar über die Kontroll-VM auf die Hardware zugreifen können. Nur Prozessor und Hauptspeicher werden direkt vom Hypervisor verwaltet und werden den VM mittels der hardware-gestützten Virtualisierung zugänglich (Abbildung 2-18).

Abbildung 2-18: Microkernel Hypervisor – Hardware-Zugriff



Quelle: Eigene Darstellung, nach [Citrix, 2008], S. 64

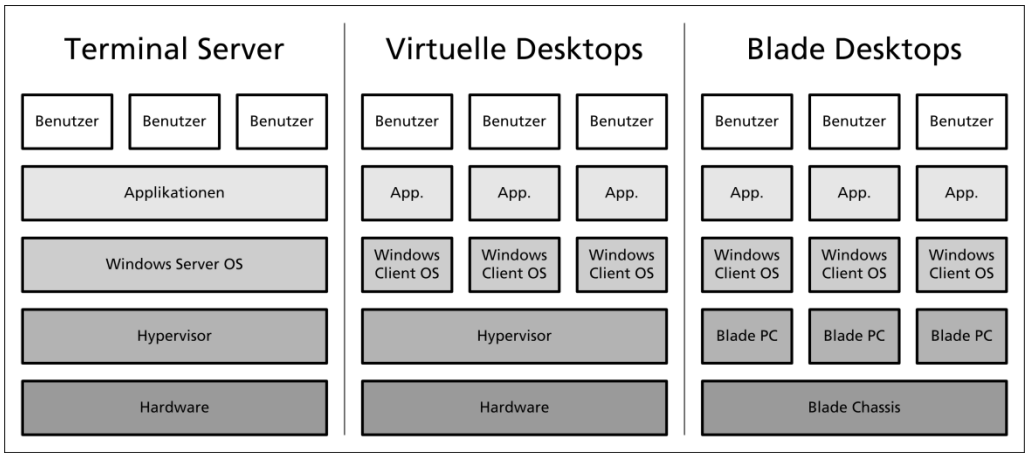
Die Kontroll-VM dient dem Erzeugen und Verwalten weiterer VM. Ebenso verwaltet sie die übrige Hardware und bindet beispielsweise Massenspeicher an, sei es eine im lokalen System verbaute Festplatte oder Speicher, der von einem SAN/NAS (vgl. Kapitel 2.4.2) bereitgestellt wird. Wenn eine der unprivilegierten VM auf entsprechende Hardware zugreifen möchte, kommuniziert sie dazu mittels der generischen paravirtualisierten Treiber mit der Kontroll-VM, die konkurrierende Zugriffe mehrerer VM verwaltet und zwischen den VM und der tatsächlichen Hardware vermittelt.

2.3 Desktop-Virtualisierung

Der Begriff der Desktop-Virtualisierung ist nicht scharf umrissen, sondern wird in der Literatur und insbesondere von Hardware- und Software-Anbietern in unterschiedlicher Weise verwendet. Grundsätzlich können bereits Terminal Server als Ausprägung virtueller Desktops verstanden werden. Denn, wie eingangs erläutert, hat die Virtualisierung in allgemeiner Form das Ziel, scheinbar vorhandene Systeme und Zustände zu simulieren. Und eben dies leistet ein Terminal Server, indem er für einen individuellen Benutzer den Eindruck erweckt, diesem Benutzer einen Desktop oder eine Anwendung exklusiv zur Verfügung zu stellen, obwohl sich der Benutzer im Hintergrund tatsächlich ein Multi-User-Betriebssystem mit vielen weiteren Anwendern teilt. Im idealen Fall bemerken die Endbenutzer dies nicht.

Jedoch sind auch heute noch zahlreiche Applikationen für die Microsoft Windows Plattform im Einsatz, die nicht nach den Programmierrichtlinien von Microsoft[34] entwickelt wurden. Ebenso gibt es Anwendungen, die auf Grund ihrer Leistungsanforderungen oder dem Zweck der Anwendung nach[35] nicht sinnvoll auf Terminal Servern zu betreiben sind.

Abbildung 2-19: Ausprägungen der Desktop-Virtualisierung



Quelle: Eigene Darstellung

Vor diesem Hintergrund fasst der Oberbegriff der Desktop-Virtualisierung heute typischerweise Technologien zusammen, die dazu dienen, statt eines Terminal Servers individuelle Client-Betriebssysteme, wie Microsoft Windows Vista oder Windows 7, im Rechenzentrum zu betreiben und diese den Endanwendern mit den Methoden der entfernten Präsentation zugänglich zu machen. Hierfür ist in der Praxis auch der Begriff des „Hosted Virtual Desktop" anzutreffen in Abgrenzung zum „Shared Virtual Desktop", den ein Terminal Server bereitstellt. Die Basis bildet in der Regel ein Typ 1 Hypervisor mit den Techniken hardware-gestützter Virtualisierung und/oder binärer Translation bzw. Paravirtualisierung. Die Endanwender können die auf dem Hypervisor laufenden Client-Betriebssysteme im Fernzugriff über Protokolle, wie beispielsweise RDP, ICA oder PC-over-IP (PCoIP)[36] von VMware, nutzen. Bei höheren Anforderungen an die Leistungsfähigkeit des zentral bereitgestellten Desktops können an

[34] Die Programmierrichtlinien für die Windows Plattform sehen vor, dass Applikationen benutzerbezogene Einstellungen nur im benutzerspezifischen Teil der Registrierung, dem Bereich HKEY_CURRENT_USER sowie in Form von Dateien nur im Benutzerprofil des jeweiligen Anwenders ablegen sollen. Halten sich Anwendungen nicht daran und versuchen, Registrierungsschlüssel im Bereich HKEY_LOCAL_MACHINE oder Dateien im Programmverzeichnis abzulegen, so führt dies zu Fehlern. Die Zugriffsrechte zu lockern, so dass auch unprivilegierte Benutzer in diese Bereiche schreiben können, ist insbesondere auf einem Multi-User-System keine Option, da dies zu Konflikten führen und die Stabilität des Servers beeinträchtigen kann.

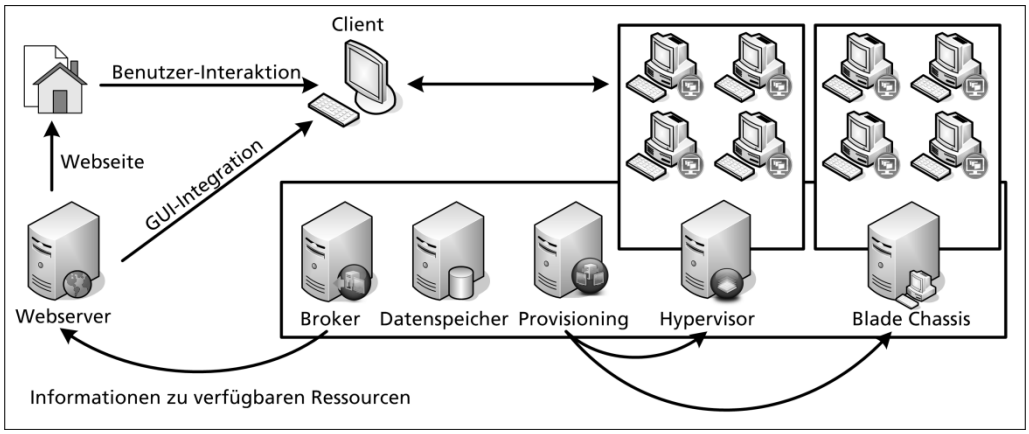
[35] Beispiele hierfür sind leistungshungrige Applikationen aus den Bereichen Desktop Publishing (DP) und Computer Aided Design (CAD) ebenso wie Anwendungsfälle aus dem Bereich der Softwareentwicklung. Wenn ein Anwender selbst Software kompilieren und testen muss, ist dies auf einem Terminal Server nicht zu empfehlen, da auch dies die Stabilität des Servers beeinträchtigen kann.

[36] Das PC-over-IP (PCoIP) Protokoll wurde ursprünglich von dem Halbleiter-Entwickler Teradici als hardware-gestützte Remote-Lösung entworfen und im Jahr 2008 von VMware lizenziert. Anschließend entstand in einer Kooperation der beiden Firmen ein Software-Implementierung, die in der Desktop-Virtualisierungslösung VMware Horizon (https://www.vmware.com/de/products/horizon-view/ (abgerufen am 01.02.2016)) zum Einsatz kommt (vgl. Kapitel 2.4.1).

Stelle von VM auf einem Hypervisor fallweise auch direkt auf einer Hardware, beispielsweise auf Blade PC (vgl. Kapitel 2.4.2), ausgeführte Client-Betriebssysteme Verwendung finden (Abbildung 2-19). Bei der Bereitstellung individueller Betriebssystem-Instanzen pro Benutzer entfällt innerhalb der VM natürlich die Multi-User-Fähigkeit, was dazu führt, dass bei intensivem Gebrauch dieser Art von virtuellen Desktops deutlich mehr VM nötig sind.

Um den damit verbundenen Verwaltungsaufwand und auch den Ressourcenbedarf zu reduzieren, haben diverse Anbieter Infrastruktur-Lösungen geschaffen, die auf Ebene des Hypervisors oder darüber ansetzen. Hierfür ist der Begriff der „Virtual Desktop Infrastructure (VDI)" verbreitet. In einer solchen Infrastruktur tritt an die Stelle der Terminal Server ein Vermittler, von manchen Anbietern auch als „Broker" oder „Controller" bezeichnet. Dieser kommuniziert die Informationen zu verfügbaren Ressourcen, in diesem Fall Desktop-Instanzen, in der Regel auf ähnlichem Weg an die Clients, wie dies auch im Fall der Terminal Server Farm geschieht (vgl. Kapitel 2.1.3.1). Im Gegensatz zur Terminal Server Infrastruktur stellen Clients allerdings keine direkte Verbindung zum Broker her. Der Broker vermittelt zwischen den Clients und diesen Desktop Instanzen, woraufhin zwischen beiden eine direkte Verbindung nach dem Prinzip der entfernten Präsentation über ein Remote-Protokoll etabliert wird (Abbildung 2-20).

Abbildung 2-20: Virtual Desktop Infrastructure



Quelle: Eigene Darstellung

Wenngleich die am Markt verfügbaren VDI-Lösungen sich in der Art und Weise, wie die Desktop-Instanzen verwaltet werden, in Nuancen unterscheiden, werden meist zwei grundlegende Typen von virtuellen Desktops unterschieden:

- **Individuelle Desktops:** Bei einem individuellen Desktop handelt es sich typischerweise um einen bereits existierenden Desktop, der in eine VDI eingebunden wird. Dies kann ein physischer Desktop-PC, ein Blade-PC oder auch eine VM sein. Charakteristisch ist in jedem Fall, dass solche Desktops separat zu pflegen sind. Zwischen einzelnen Desktops können entsprechend bezüglich der Hard- und Softwareausstattung größere Abweichungen bestehen, was es auf der einen Seite ermöglicht, die individuellen Anforderungen einzelner Endanwender gezielt zu bedienen. Auf der anderen Seite bringt ein solches Herangehen

natürlich den größten Verwaltungsaufwand sowie Ressourcenbedarf mit sich. Oftmals werden individuelle Desktops ihren Benutzern in einer 1:1 Beziehung fest zugeordnet.

- **Desktop-Pools:** Ein Pool fasst gleichartige Desktops zu einer Gruppe zusammen. Benutzer werden bei der Verbindungsaufnahme in einer n:m Beziehung mit einem beliebigen Desktop aus der Gruppe verbunden. In der Regel wird der Desktop, sobald der Benutzer sich wieder abmeldet, auf den ursprünglichen Zustand zurückgesetzt, was den Verwaltungsaufwand reduziert. Andererseits bietet diese Variante weniger Freiheitsgrade in der Anpassung der Arbeitsumgebung durch die Anwender als ein individueller Desktop.

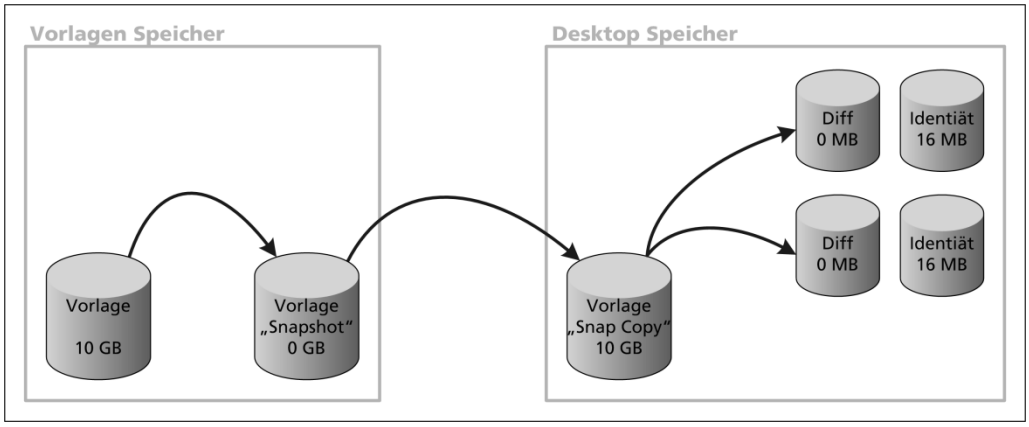
2.3.1 Provisionierung von Desktops

Die Verwaltung von Desktops in einem Pool, die wahlweise auf einem Hypervisor oder direkt in Hardware ausgeführt als Blade PC betrieben werden, übernimmt die Rolle des Brokers, bei der es sich auch um einen separaten Server handeln kann. Hierbei kommen verschiedene Techniken zum Einsatz, um aus einem oder möglichst wenigen Vorlagen automatisiert eine größere Menge Desktops zu erstellen. Für eine solche Vorlage ist auch die Bezeichnung „Golden Image" gebräuchlich, der Vorgang selbst wird als Provisionieren, engl. Provisioning, bezeichnet. Beim Provisionieren können zwei Arten der Umsetzung unterschieden werden – eine Variante, die im Speichersubsystem des Hypervisors ansetzt und eine weitere, die Netzwerkdienste mit einbezieht.

Den ersten Ansatz verfolgen Lösungen wie die sogenannten Linked Clones der VDI-Lösung VMware Horizon oder auch die Machine Creation Services (MCS), eine Komponente von Citrix XenDesktop. „Die MCS verwenden eine virtuelle Maschine als Vorlage, von der sie einen Snapshot erstellen und auf einen separaten Storage-Bereich kopieren (…) Die Snap Copy dient als Basis für beliebig viele gleichartige Desktops. Dies wird möglich, indem der eigentliche Snapshot als Master Image schreibgeschützt abgelegt wird. Sämtliche VMs erhalten zwei weitere Disks – eine Diff Disk, in der sämtliche Änderungen am Master Image abgelegt werden, sowie eine Identity Disk, die pro Maschine persistent deren Identität im Active Directory speichert. Gegenüber den VMs erscheinen diese Disks transparent als eine Festplatte." (vgl. [Knermann, 2011a])

Der Vorteil dieser Herangehensweise ist, dass eine große Anzahl virtueller Desktops bereitgestellt werden kann, ohne dass in entsprechendem Umfang persistent Platz im Speichersubsystem des Hypervisors alloziert werden müsste (Abbildung 2-21). Die Vorlage wird lediglich einmal in schreibgeschützter Form vorgehalten und es müssen pro virtuellem Desktop nur Informationen zur eindeutigen Identität der VM, in einer Windows Domäne insbesondere zum Active Directory Konto, sowie sämtliche Abweichungen von der Vorlage, d. h. Schreibzugriffe im laufenden Betrieb, als Differenz festgehalten werden. Initial ist diese Differenz noch nicht vorhanden, wächst erst im Laufe des Betriebs an und wird typischerweise beim Abmelden eines Benutzers wieder verworfen, so dass der Platzbedarf der virtuellen Desktops in diesem Betriebsmodell gering ist.

Abbildung 2-21: Verwaltung virtueller Desktops im Speichersubsystem



Quelle: Eigene Darstellung

Alternativ zum Provisionieren der virtuellen Desktops im Speicher des Hypervisors existieren Ansätze, die zusätzliche Dienste auf Netzwerkebene mit einbeziehen. Ein Beispiel für die Umsetzung findet sich mit den Citrix Provisioning Services (PVS), deren Funktionsprinzip auf einer als „Disk Streaming" bezeichneten Technologie basiert, welche Citrix bereits Ende 2006 mit der Firma Ardence zugekauft und in die eigenen Produktlinien integriert hat. Das Streaming ermöglicht es, sowohl physische als auch virtuelle Maschinen von einem über das Netzwerk bereitgestellten Image an Stelle einer lokalen Festplatte zu starten. Ein Provisioning Server kann viele verschiedene Images als sogenannte vDisks in einem Store[37] vorhalten und ordnet diese anhand der MAC-Adresse seinen Clients, den sogenannten Target Devices, zu.

Die Provisioning Services sind hierachisch strukturiert und verwenden ähnlich zu den Terminal Servern den Begriff der Farm als oberste Verwaltungseinheit, die ein oder mehrere Server enthalten kann. Die Konfiguration der Farm wird zentral in einer Datenbank abgelegt. Eine Farm untergliedert sich in ein oder mehrere Sites, die wiederum Provisioning Server, Gruppen von Images, vDisk Pool genannt, sowie Gruppen von Target Devices, als Device Collection bezeichnet, enthalten (vgl. [Knermann, 2010b). Zusätzlich zu den Provisioning Services sind in der Regel auch DHCP- und TFTP-Dienste erforderlich, da die Target Devices nicht zwingend über lokalen Massenspeicher verfügen müssen und stattdessen vom TFTP-Server lediglich ein Boot-Programm laden, das seinerseits die Verbindung zu einer der vDisks auf dem Provisioning Server herstellt. Hierfür bieten sich die Alternativen, die Clients über den heute kaum mehr gebräuchlichen BOOTP-Dienst oder das Preboot Execution Environment (PXE) zu starten. Diese

[37] „Bei einem Store handelt es sich um einen logischen Bezeichner für den physischen Ablageort von vDisks. Dieser lässt sich der kompletten Farm oder einer Site zuordnen. Neben der lokalen Festplatte eines PVS Servers kommen CIFS-Freigaben oder auch anderweitige SAN-/NAS-Lösungen in Betracht. (…) Das Boot-Image für ein oder mehrere Target Devices wird als vDisk im Store bereitgestellt. Hierbei kommt das Microsoft Virtual Hard Disk (VHD)-Format zum Einsatz. (…) Beim Anlegen einer vDisk kann deren Größe als ‚Fixed' oder ‚Dynamic' konfiguriert werden. Im ersten Fall wird sofort der komplette angegebene Speicherplatz belegt, die zweite Option lässt die vDisk dynamisch bis zur maximalen Größe anwachsen. Dies spart Speicherplatz, muss aber auch mit Bedacht eingesetzt werden, da damit das Überbuchen des tatsächlich verfügbaren Platzes möglich ist." (vgl. [Knermann, 2010b]).

Variante wird heute in der Regel von allen Netzwerkkarten und insbesondere auch den virtuellen Netzwerkadaptern der aktuellen Typ 1 Hypervisoren unterstützt. Die Adresse des TFTP-Servers sowie der Pfad zum Boot-Programm können den Clients über entsprechende DHCP-Optionen[38] bekannt gegeben werden. Das Boot-Programm wird vom Client ausgeführt und nimmt Kontakt zum Provisioning Server auf, der dem Client anhand seiner MAC-Adresse eine vDisk zuweist, von der der Bootvorgang anschließend über das Netzwerk[39] fortgesetzt wird.

Die Provisioning Services verwalten die Identitäten der Clients, innerhalb einer Windows Domäne insbesondere deren AD Konten, und bieten Unterstützung beim Ausbringen von Updates in der Form, dass es im laufenden Betrieb möglich ist, eine aktualisierte Version einer vDisk vorzubereiten und Clients nur durch einen Neustart mit dieser neuen vDisk-Version zu verbinden (vgl. [Knermann, 2010c]).

Als Nachteil dieses Ansatzes ist festzuhalten, dass das Provisionieren über das Netzwerk mehr Server und Dienste involviert als eine der Varianten, die nur im Speicher des Hypervisors arbeiten. Der große Vorteil ist andererseits aber, dass ein netzwerkorientierter Ansatz nicht auf virtuelle Maschinen als Clients beschränkt ist. Vielmehr lassen sich sowohl VM als auch physische Ziele, z. B. Blade PC, in gleicher Weise bedienen. Dies ist nicht allein auf das Ausführen von Client-Betriebssystemen beschränkt, sondern kann beispielsweise auch eingesetzt werden, um innerhalb einer Terminal Server Farm viele gleichartige Systeme bereitzustellen und deren Update-Prozess zu vereinheitlichen (vgl. [Knermann, 2010d]). Das Provisionieren von Systemen in seinen verschiedenen Ausprägungen kann damit als Basistechnologie zur Bereitstellung von Anwendungen sowohl auf Terminal Servern als auch auf virtuellen Desktops verstanden werden.

Eine umfangreiche Übersicht der derzeit verfügbaren Lösungen für die Bereitstellung virtueller Desktops – über die bisher beispielhaft erwähnten Möglichkeiten hinaus – findet sich in [Spruijt, 2013]. Im Folgenden werden Infrastrukturen basierend auf Citrix XenDesktop 7.x im Vordergrund stehen.

[38] Die IP-Adresse eines TFTP-Servers kann mit der DHCP-Option 66 übermittelt werden. Die DHCP-Option 67 enthält den Dateinamen des Boot-Programms. Die Provisioning Server verwenden ein Boot-Programm mit dem Namen 'ARDBP32.BIN' (vgl. http://support.citrix.com/article/CTX115094 (abgerufen am 01.02.2016)).

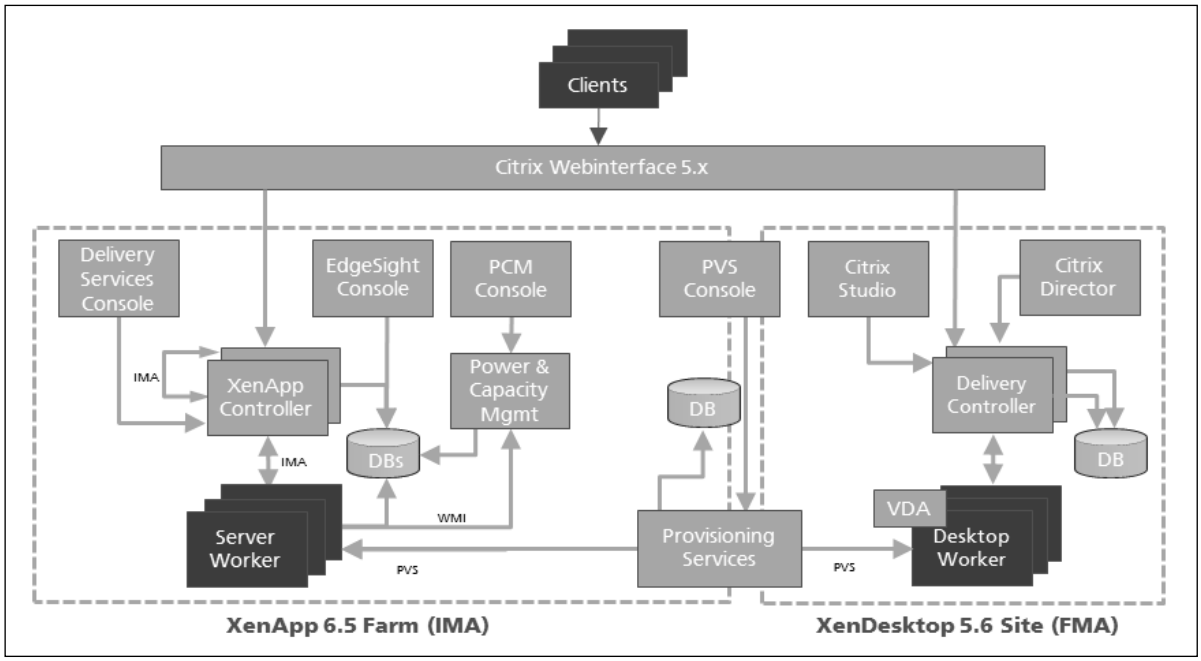
[39] „Anders als der Begriff ‚Streaming' vielleicht implizieren mag, wird die vDisk (…) nicht auf das Target Device heruntergeladen – entsprechend ist keine lokale Festplatte erforderlich. Möglich macht dies ein spezieller Festplatten-Treiber, der dem Betriebssystem die über das Netz verbundene vDisk als lokales Medium präsentiert. Im Hintergrund kommuniziert der Treiber mit dem PVS Server über ein proprietäres Protokoll auf Basis von UDP. Diese Technik arbeitet also ähnlich iSCSI, allerdings mit dem signifikanten Unterschied, dass nicht ein bereits bestehendes Protokoll in eine IP-Verbindung verpackt wird, sondern das Protokoll von vornherein für den Netzwerkeinsatz entwickelt wurde. (…) Geht während eines Boot-Vorgangs oder im laufenden Betrieb die Verbindung verloren, so führt dies nicht zu einem Absturz der Target Devices. Diese bleiben stattdessen stehen und laufen weiter, sobald die Verbindung wiederhergestellt ist." (vgl. [Knermann, 2010b]).

2.3.2 Citrix XenDesktop 7.x

Wie bereits in Kapitel 2.1.3.2 dargelegt, existierten die Terminal Server Lösung XenApp und die Infrastruktur für virtuelle Desktops XenDesktop aus der historischen Entwicklung heraus als komplett separate Systeme, wobei frühe Versionen von XenDesktop mit IMA zunächst auf das gleiche Protokoll und auch auf ein ähnliches Funktionsprinzip wie die Terminal Server Farmen setzten. Server in der Rolle von Desktop Delivery Controllern (DDC) fungierten dabei als zentraler Verwaltungspunkt, während an die Stelle der Worker einzelne – wahlweise physische oder virtuelle – Instanzen von Desktop-Betriebssystemen traten.

Bei XenDesktop hatte Citrix die IMA als Protokoll für die interne Kommunikation bereits durch die Flexcast Management Architechture (FMA) ersetzt, da IMA zwar auf mehrere Hundert Terminal Server skalierbar war, nicht aber auf Tausende oder gar Zehntausende von Desktops. XenDesktop brachte natürlich auch die Anforderung für eine zusätzliche Datenbank sowie eigene Management-Werkzeuge mit sich, nämlich das Desktop Studio für die grundlegende Konfiguration der Site und den Desktop Director als webbasiertes Frontend für typische Helpdesk-Aufgaben im laufenden Betrieb.

Abbildung 2-22: Paralleler Betrieb von XenApp 6.5 und XenDesktop 5.6



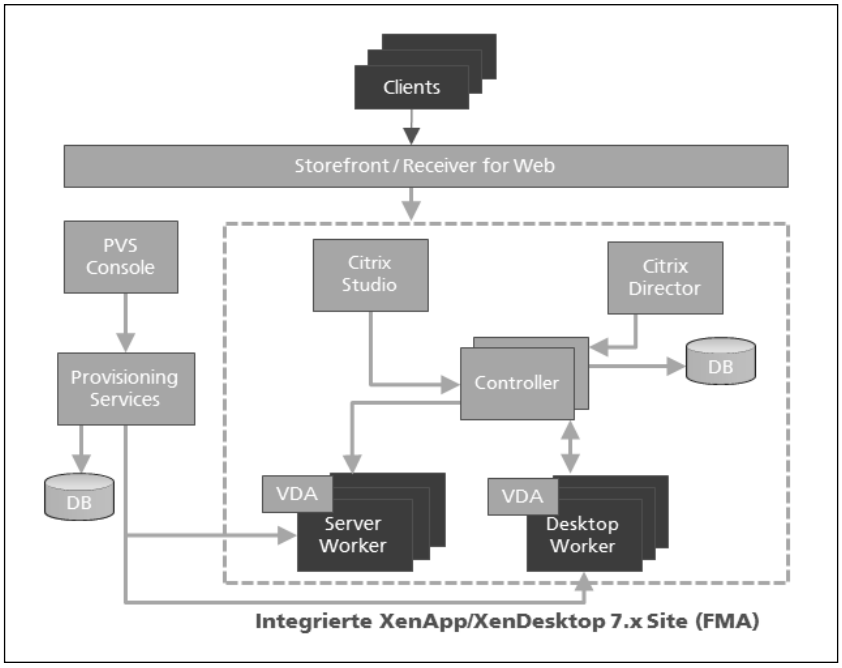
Quelle: Eigene Darstellung

Sollten in einer Betriebsumgebung mit existierenden Terminal Servern virtuelle Desktops auf Basis von XenDesktop 5.6 hinzugefügt werden, bedeutete dies, dass neben der XenApp Farm eine eigenständige XenDesktop Site gepflegt werden musste. Zwar ließ sich die Komplexität vor dem Endanwender verstecken, indem die Ressourcen beider Welten in einem gemeinsamen Webinterface, genannt StoreFront, zusammengefasst wurden. StoreFront erforderte aber eine weitere Datenbank-Instanz. Wer als Administrator mit XenApp 6.5 und

XenDesktop 5.6 das komplette Portfolio der Anwendungsbereitstellung erschließen wollte, sah sich somit einer kaum überschaubaren Vielzahl von Management Werkzeugen und Datenbanken gegenüber (Abbildung 2-22), ganz zu schweigen von dem erheblichen Aufwand bei der Ersteinrichtung einer solchen Infrastruktur.

Um diesen Gordischen Knoten zu lösen, hat Citrix mit XenDesktop 7.x sämtliche beteiligten Produkte renoviert. XenDesktop 7.x führt die Entwicklungszweige von XenDesktop und XenApp konsequent zusammen, was besonders für XenApp die größten Umwälzungen bedeutet (vgl. [Knermann, 2013]). Auch die Terminal Server kommunzieren nun intern über die FMA. Dies bedeutet, dass nunmehr ein und dieselben Controller für beide Welten zuständig sind und nur noch ein Paket des Virtual Desktop Agent (VDA) für die Installation auf den Workern erforderlich ist – unabhängig davon, ob es sich bei dem Worker beispielsweise um einen Windows 8.1 Desktop oder einen Windows Server 2008 R2 Terminal Server handelt (Abbildung 2-23).

Abbildung 2-23: Integrierter Betrieb von XenApp/XenDesktop 7.x



Quelle: Eigene Darstellung

Die Setup-Routine des VDA integriert zudem weitere Komponenten, die bisher separat zu pflegen waren. So sind nun der Agent für das Monitoring per EdgeSight und auch das Citrix Profile Management gleich mit enthalten. Welche Funktionen nutzbar sind, entscheidet sich anhand der lizenzierten Edition. Entsprechend ist für eine solche Umgebung nur noch eine Datenbank erforderlich, die wahlweise auf einem SQL Server 2008 R2 SP2 oder einem SQL

Server 2012 SP1 laufen darf[40]. Im einfachsten Fall installiert das Controller-Setup die Express Variante des SQL Server 2012 einfach mit.

Für Installationen mit mehr als einem Controller empfiehlt es sich natürlich, die Datenbank auf einen eigenständigen Datenbankserver auszulagern. Da die FMA erwartet, dass die Datenbank dauerhaft online ist, sollte dieser Server tunlichst hochverfügbar auslegt sein, wobei XenDesktop sämtliche von Microsoft gebotenen HA-Techniken unterstützt.

Mit XenApp/XenDesktop 7.6 werden zudem die „StoreFront"-Dienste bereits auf Version 3.0 aktualisiert. Es handelt sich dabei um eine komplett neu entwickelte web-basierte Lösung für den Zugriff auf die Ressourcen einer Betriebsumgebung. StoreFront löst damit das bisherige Citrix Webinterface ab. StoreFront 3.0 benötigt keine separate SQL Datenbank mehr, sondern speichert die Einstellungen der User in XML-Dateien. Sollen mehrere StoreFront Server zum Einsatz kommen, werden diese Dateien automatisch synchronisiert, was die Komplexität des Aufbaus deutlich reduziert.

XenDesktop 7.6 FP2 und höher erweitern die Infrastrukturlösung zudem um die Möglichkeit, auf Seiten der Virtualisierung neben Windows-Betriebssystemen auch Linux-Desktops als veröffentlichte Ressourcen bereitzustellen (vgl. [Knermann, 2015]).

2.3.3 Applikations-Virtualisierung

Auch bei der Applikations-Virtualisierung oder Anwendungsvirtualisierung handelt es sich nicht um eine klar abgegrenzte Technik, sondern um einen Oberbegriff, unter dem mehrere Anbieter jeweils unterschiedliche Ansätze verfolgen, die aber im Kern einem ähnlichen Ziel dienen. Dabei geht es darum, Anwendungen nicht wie üblich direkt auf einem Betriebssystem zu installieren, sondern diese vom unterliegenden Betriebssystem zu abstrahieren und zu isolieren. Anwendungen haben dabei zwar weiterhin die Möglichkeit, lokale Ressourcen, also Teile des Betriebssystems, DLL oder auch andere Anwendungen, zu nutzen. Die Anwendungen selbst werden aber innerhalb einer virtuellen Umgebung, oftmals auch als Sandbox bezeichnet, gekapselt und haben ihrerseits nicht die Möglichkeit, direkt in das Betriebssystem, d. h. auf die lokale Festplatte oder im Fall eines Windows-Systems in die Registrierung, zu schreiben.

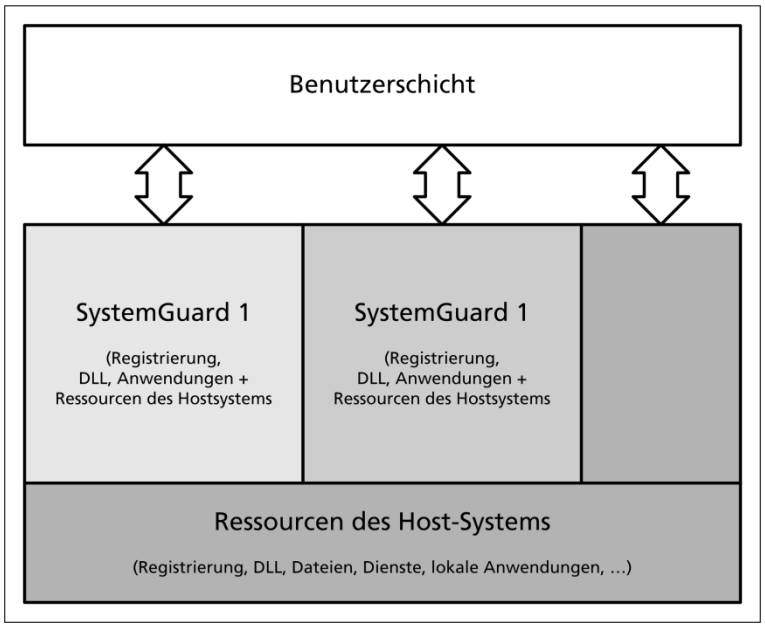
Die Motivation für eine solche Herangehensweise besteht darin, Konflikte zwischen Anwendungen sowie zwischen Anwendung und Betriebssystem zu vermeiden. So kann eine Anwendung, die innerhalb einer Sandbox installiert ist, durch Löschen der entsprechenden Sandbox rückstandsfrei und ohne aufwändige Deinstallation wieder vom ausführenden Computer entfernt werden. Weiterhin ist es auf diese Weise möglich, verschiedene Versionen einer Anwendung parallel auf einem Computer zu nutzen, beispielsweise Microsoft Office 2010 und 2013. Des Weiteren kann das Isolieren helfen, Anwendungen im Multi-User-Betrieb auszuführen, die nicht nach den Microsoft Programmierrichtlinien entwickelt wurden und

[40] XenDesktop 7.6 Systemvoraussetzungen: http://docs.citrix.com/en-us/xenapp-and-xendesktop/7-6/xad-system-requirements-76.html (abgerufen am 01.02.2016).

entsprechend direkt installiert nicht auf einem Terminal Server lauffähig wären. So ist es innerhalb der Sandbox möglich, dass eine im Kontext eines unprivilegierten Benutzers laufende Anwendung *virtuell* in das Programmverzeichnis oder in den maschinen-spezifischen Teil der Windows-Registrierung schreibt, obwohl der Anwender *tatsächlich* dort keine Schreibrechte besitzt. Die Sandbox fängt in diesem Fall die Schreibzugriffe ab und leitet diese in das Benutzerprofil oder den Benutzer-spezifischen Teil der Windows-Registrierung um.

Die vormals als Application Isolation Environment (AIE) bezeichnete und inzwischen eingestellte Anwendungsvirtualisierung aus dem Hause Citrix arbeitet in dieser Weise (vgl. [Knermann, 2007]) und ähnlich verhält es sich auch mit der entsprechenden Lösung von Microsoft namens App-V[41], welche im Folgenden als Beispiel für das Funktionsprinzip dienen soll. So wird im Falle von App-V grundsätzlich jede Anwendung in einer eigenen Sandbox, dem sogenannten SystemGuard, gekapselt (Abbildung 2-24). Es ist allerdings auch möglich, mehrere Anwendungen gemeinsam in einem SystemGuard zu paketieren.

Abbildung 2-24: Applikationsvirtualisierung mittels Microsoft App-V



Quelle: Eigene Darstellung, nach [Lüdemann, 2007]

Hierzu ist ein System erforderlich, das dazu dient, eben diese Pakete zu erstellen. Das System sollte möglichst der Zielplattform entsprechen und führt eine Software – im Beispiel von Microsoft als Sequencer und von Citrix als Profiler bezeichnet – aus, welche die Sandbox

[41] Bei Microsoft App-V handelt es sich um das ehemalige Softricity SoftGrid, welches von Microsoft zugekauft und in das eigene Lösungsportfolio integriert wurde. App-V wird nicht als eigenständiges Produkt vermarktet, sondern als Teil des Microsoft Desktop Optimization Packs (MDOP) oder als „App-V for RDS" integriert in die Lizenzierung der Remotedesktopdienste. Citrix verfuhr mit der hauseigenen Anwendungsvirtualisierung in ähnlicher Weise. Diese war Bestandteil diverser Editionen von XenApp und XenDesktop.

bereitstellt, in der sämtliche Ressourcen der Anwendung, d. h. Dateien und Registrierungseinträge, gespeichert werden. Soweit die Anwendung lesenden Zugriff auf im unterliegenden Betriebssystem vorhandene Objekte benötigt, wird dieser Zugriff transparent gewährt. Versucht die Anwendung dagegen schreibenden Zugriff, um beispielsweise eine im System-Ordner des Betriebssystems vorhandene DLL durch eine andere Version zu ersetzen, so wird dieser Zugriff in die virtuelle Umgebung der Anwendung umgeleitet.

Auf den Clients der Anwendungsvirtualisierung – hierbei kann es sich um ein lokal installiertes System, einen Terminal Server oder einen virtuellen Desktop handeln – wird als Gegenstück zum Sequencer ein Agent benötigt, welcher in der Lage ist, das Paket mit der virtualisierten Anwendung zu laden, dieses in einer sicheren Umgebung auszuführen und weiterhin im laufenden Betrieb Lese- und Schreibzugriffe zu virtualisieren. Der Prozess der Übermittlung der Applikationspakete von einem zentralen Speicherort zum ausführenden Client wird als Streaming bezeichnet. „Dabei werden die Daten meist per SMB-Protokoll übertragen. Für Benutzer, die sich außerhalb des Unternehmensnetzwerkes befinden, bieten einige Hersteller die Möglichkeit einer Übertragung auf HTTP/HTTPS-Basis an." (vgl. [Vogel, 2010], S. 15).

Das Streaming kann je nach Lösung auf Basis einzelner Dateien, bevorzugt aber blockweise erfolgen. So folgen App-V wie auch das Anwendungsstreaming von Citrix letzterem Ansatz. Die ausführbaren Dateien und Ressourcen der Anwendung werden dabei im Streaming-Paket so organisiert, dass die Anwendung bereits gestartet werden kann, bevor das komplette Paket übertragen ist. Je nach Anwendung müssen nur 10-30 Prozent des gesamten Pakets geladen werden, bevor die Anwendung startet (vgl. [Spruijt, 2013a], S. 22). Dies und die Möglichkeit, bereits geladene Anwendungspakete in einem Cache lokal zwischen zu speichern, reduziert die Startzeit ebenso wie die Belastung des Netzwerkes.

Nichtsdestotrotz ist zu berücksichtigen, dass sich der herkömmliche Zugriff auf Anwendungen im Rahmen der entfernten Präsentation und das Anwendungsstreaming direkt an Endgeräte in der Art und Weise der Netzwerknutzung grundsätzlich unterscheiden. So benötigt eine Sitzung auf einem Terminal Server oder virtuellen Desktop nur eine schmalbandige Verbindung zur Übertragung des jeweiligen Remote-Protokolls. Diese muss allerdings kontinuierlich zur Verfügung stehen. Im Falle der Anwendungsvirtualisierung muss nur initial – zumindest, bis alle benötigten Komponenten geladen sind – eine Verbindung zum Speicherort der Anwendungspakete bestehen. Je nach Anwendung muss dabei aber in kurzer Zeit eine im Vergleich zu den Remote-Protokollen signifikant größere Datenmenge übertragen werden. Daher sind die Konzepte der Anwendungsvirtualisierung am effektivsten einzusetzen innerhalb von breitbandigen lokalen Netzen, zu denen dauerhaft eine Verbindung besteht.

Sollen auch mobile Clients angebunden werden, die sich zeitweise außerhalb des lokalen Netzes befinden und beispielsweise über Einwahlverbindungen nur diskontinuierlich eine Verbindung zum Speicherort der Anwendungspakete unterhalten, so besteht zum einen die Möglichkeit, auf diesen Clients den Cache bereits mit Anwendungspaketen zu befüllen, bevor diese das lokale Netz verlassen. Alternativ ist ein mehrstufiges Betriebsmodell denkbar, in dem die Pakete virtueller Anwendungen nicht direkt per Streaming an ein Endgerät übertragen

werden, sondern mittelbar zunächst an einen Terminal Server oder virtuellen Desktop übertragen, dort ausgeführt und nach dem Prinzip der entfernten Präsentation über ein Remote-Protokoll an das Endgerät übertragen werden. Natürlich ist in diesem Fall eine kontinuierliche Netzwerverbindung zwischen Client und dem ausführenden Terminal Server oder Desktop nötig. Der Vorteil ist indes, dass diese Verbindung schmalbandig ausgeführt sein darf und die Anwendung somit beispielsweise auch über Mobilfunkverbindungen nutzbar wird.

Ein weiterer signifikanter Vorteil ergibt sich für den Betrieb von Terminal Servern und virtuellen Desktops, insbesondere wenn diese, wie im vorherigen Kapitel erläutert, provisioniert werden und dies mit Methoden der Anwendungsvirtualisierung kombiniert wird. So ist es auch bei divergierenden Anforderungen seitens der Endanwender nicht mehr zwingend erforderlich, je Anwendergruppe oder gar pro einzelnem Benutzer, ein separates Image vorzuhalten. Stattdessen wird es möglich, lediglich ein Basisimage zu erstellen, das beispielsweise von allen Benutzern benötigte Komponenten – wie einen Virenscanner, ein Office Paket sowie den Agenten der Anwendungsvirtualisierung – enthält, während spezielle, nur von einzelnen Benutzern gewünschte Applikationen fallweise zur Laufzeit per Anwendungsvirtualisierung und -streaming hinzugeladen werden. Dies vereinfacht das Imagemanagement der Terminal Server und Desktops und erlaubt dennoch die Individualisierung der Arbeitsumgebung.

Alternativ zu den beispielhaft erwähnten Lösungen, die einen Agenten auf dem Clientsystem erfordern, existieren zudem Ansätze, wie z. B. VMware ThinApp, die ohne einen Client-seitigen Agenten auskommen. In diesem Fall wird, neben den Dateien und Ressourcen der zu virtualisierenden Anwendung, auch die Laufzeitumgebung der jeweiligen Lösung mit paketiert, so dass ein Paket entsteht, welches autark und ohne weitere Komponenten zu installieren, auf einem Zielsystem ausführbar ist. Dies ist von Vorteil, wenn es sich bei dem Ziel um einen Desktop handelt, auf dem der jeweilige Endanwender als alleiniger Benutzer arbeitet. Für Terminal Server empfiehlt sich dies dagegen typischerweise nicht, sofern die Sicherheitsmaßnahmen so gewählt sind, dass Benutzer nicht beliebige Programme, insbesondere von Speichorten im Netzwerk, ausführen dürfen (vgl. Kapitel 3.3.1.2 und 3.3.1.3). Eine umfangreiche Übersicht der derzeit verfügbaren Lösungen für die Virtualisierung von Anwendungen – über die zuvor erwähnten Möglichkeiten hinaus – findet sich in [Spruijt, 2013a].

2.3.4 Client-Hypervisoren

Der Begriff des Client-Hypervisors bezeichnet keine weitere, eigenständige Technologie, sondern im Kern die Verwendung der bereits vorgestellten Ansätze mit einem bestimmten Fokus, nämlich der parallelen Ausführung mehrerer VM lokal auf einem Client-Computer. Auch bei der Virtualisierung auf einem Client werden Typ 1 und Typ 2 Hypervisoren unterschieden (vgl. [Spruijt, 2013], Abs. 3.15). Wie bereits erwähnt, kommen Typ 2

Hypervisoren bereits seit längerer Zeit typischerweise auf Client-Computern zum Einsatz[42], bringen aber den Nachteil mit sich, dass sie ein unterliegendes Betriebssystem als Hostumgebung benötigen, was zu Lasten der Performanz und Sicherheit geht (Kapitel 2.2.2). Entsprechend wurden auch Client-seitig Typ 1 Hypervisoren mit den bereits vorgestellten Techniken der hardware-gestützten Virtualisierung (Kapitel 2.2.7) in Verbindung mit Paravirtualisierung (Kapitel 2.2.6) entwickelt. Vorreiter dieses Ansatzes sind Lösungen, die auf Derivaten des Hypervisors Xen aufsetzen, so beispielsweise Qubes OS[43] und die frühere Lösung Virtual Computer NxTop basierend auf der Open Source Variante Xen sowie der Citrix XenClient, welcher sich die Code-Basis mit dem Citrix XenServer teilt. Zwischenzeitlich wurde Virtual Computer NxTop von Citrix übernommen und mit der hauseigenen Entwicklung XenClient zu einer Lösung verschmolzen (vgl. [Knermann, 2014]).

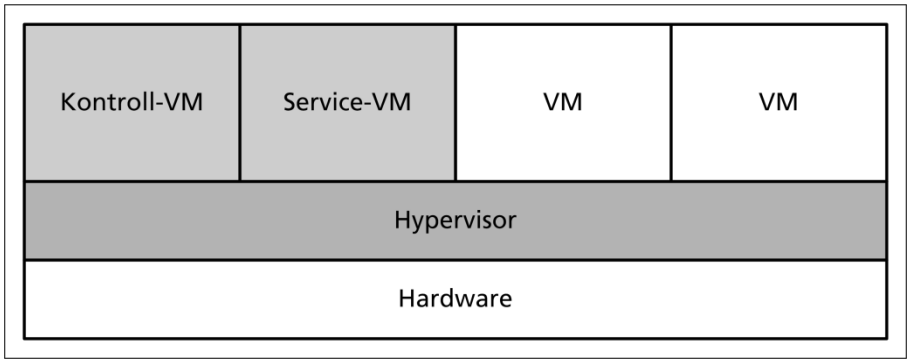
Allen Client-Hypervisoren gemein ist der Ansatz, dass im Betrieb von Rechenzentren etablierte Technologien auf die lokalen Systeme von Endanwendern übertragen werden. „Dies erfordert neben dem Schaffen einer einfach zu bedienenden Benutzeroberfläche, auch das Einbeziehen von USB-Schnittstellen, Grafikhardware oder Technologien zum Verlängern der Akkulaufzeiten in das Virtualisierungskonstrukt." (vgl. [Vogel, 2010], S. 12)

Wird ein Hypervisor wie Xen(Server) auf einem Client-Computer, beispielsweise einem Notebook, eingesetzt, so müssen innerhalb der Kontroll-VM Treiber für zusätzliche Hardwarekomponenten, wie WLAN- und USB-Chipsätze, vorhanden sein, die auf entsprechende paravirtualisierte Treiber innerhalb der Gast-VM abgebildet werden müssen. Betriebssysteme innerhalb von VM können auf diese Weise weitestgehend von der unterliegenden Hardware abstrahiert werden, was es analog zum Betrieb von virtuellen Servern im Rechenzentrum auf einfache Weise ermöglicht, VM von einer Hardware auf eine andere zu übertragen.

Werden Hypervisoren im Rechenzentrum meist per Kommandozeile oder über eine externe Managementsoftware administriert, sind diese Varianten bezogen auf einen Client-Computer in der Regel keine Alternativen. So muss das Management eines Clients so einfach gehalten und intuitiv bedienbar sein, dass es auch für weniger IT-affine Endanwender bedienbar ist. Zudem muss der Client, insbesondere wenn es sich um ein mobiles Gerät handelt, autark lauffähig sein und darf nicht von einer externen, separat zu pflegenden Software abhängen. Im Falle des Citrix XenClient tritt dazu neben die Kontroll-VM (Dom0) eine weitere virtuelle Maschine als Service-VM (Abbildung 2-25). Diese wird mit dem Hypervisor und der Kontroll-VM automatisch gestartet und bietet eine lokale grafische Benutzeroberfläche. Über diese Oberfläche erfolgen die Konfiguration des Hypervisors und der Kontroll-VM sowie die Verwaltung der Gast-VM.

[42] Verbreitete Typ 2 Client-Hypervisoren sind beispielsweise Microsoft VirtualPC, Oracle VirtualBox, VMware Workstation oder auch VMware Fusion sowie der Parallels Desktop.
[43] Qubes OS: http://qubes-os.org (abgerufen am 01.02.2016).

Abbildung 2-25: Typ 1 Client-Hypervisor



Quelle: Eigene Darstellung, nach [Citrix, 2010], S. 4

In ähnlicher Weise verfahren ebenfalls die auf Xen basierenden Client-Hypervisoren, wobei das Qubes OS einen besonderen Fokus auf die Sicherheit der Plattform legt. Die lokale grafische Management-Oberfläche startet in diesem Fall zwar direkt aus der Dom0 heraus, es kommen zusätzlich aber zwei Service-VM zum Einsatz, eine „Network Domain" zur Kapselung des Netzwerk-Stacks und eine weitere „Storage Domain" zur Anbindung und Verwaltung von Massenspeicher (vgl. [Rutkowska, 2010], S. 8 ff.).

Letzten Endes konnte sich der Citrix XenClient jedoch nicht als Produkt am Markt behaupten und die Entwicklung wurde im Oktober 2015 mit einem Support-Ende zum Dezember 2016 eingestellt. Gründe hierfür waren, zum einen die typischerweise bessere und schnellere Treiber-Unterstützung neuer Notebook- und Desktop-Hardware durch ein direkt installiertes Betriebssystem und zum anderen die immer leistungsfähigeren Prozessoren und Chipsätze neuerer Client-Computer, die die Leistungseinbußen durch den Einsatz eines Typ 2 Hypervisors relativieren. Damit wurde der Typ 1 Hypervisor XenClient zu Gunsten des Typ 2 Hypervisors Citrix DesktopPlayer[44] aufgegeben und es verbleibt das experimentelle Qubes OS als dedizierter Client-Hypervisor.

2.4 Hardware

Ob Clients auf der einen und Serversysteme auf der anderen Seite sich eignen, um als Teil einer SBC Infrastruktur Applikationen bereitzustellen, ist primär eine Frage der Software. So muss ein Endgerät unabhängig von der konkreten Ausprägung der Hardware lediglich über ein Clientprogramm für ein Remote-Protokoll, wie z. B. RDP oder ICA, verfügen. Ein Serversystem im Rechenzentrum muss die entsprechende Serverkomponente des jeweiligen Protokolls ausführen. Falls Hardware-gestützte Virtualisierung als Basis eingesetzt werden soll, müssen die Prozessoren der Server mit passenden Erweiterungen, also Intel VT-x oder AMD-V, ausgestattet sein (vgl. Kapitel 2.2.7).

[44] Citrix DesktopPlayer: https://www.citrix.de/products/desktopplayer/overview.html (abgerufen am 01.02.2016).

Sofern diese Voraussetzungen erfüllt sind, ist grundsätzlich jede Hardware für den Einsatz im Kontext von SBC geeignet. Darüber hinaus sind mit diesem Fokus aber einige Entwicklungen von besonderem Interesse, die im Folgenden untergliedert auf die Client- und die Server-Seite erläutert werden.

2.4.1 Thin Clients

Als Alternative zu einem herkömmlichen Personal Computer bieten sich heutzutage als „Thin Clients" bezeichnete Geräte an. Diese Endgeräte sind in der Regel wesentlich kleiner als ein PC (Abbildung 2-26) und verfügen typischerweise über keinerlei bewegliche Baugruppen, wie aktive Lüfter oder Festplatten.

Das Konzept des Thin Clients wurde Mitte der 1990er Jahre von einem Konsortium bestehend aus den Firmen Oracle, Apple, IBM, Netscape und SUN Microsystems geprägt. Dem bereits früher von John Burdette Gage bei SUN formulierten Motto „The network is the computer" folgend, entstand zunächst der Begriff des Network Computers (NC), der im Network Computer Reference Profile (NCRP) näher spezifiziert wurde (vgl. [IBM, 1997], S. 257 ff.). Das NCRP sah ein Endgerät ohne persistenten, lokalen Massenspeicher vor, welches gängige Internet-Standards, wie HTML, HTTP, SMTP, IMAP und insbesondere die Java-Plattform unterstützen sollte. Motivation war, ein Endgerät zu schaffen, das günstiger und einfacher zu verwalten ist als ein herkömmlicher, als „Fat Client" bezeichneter, Personal Computer, während die Ausführung der Anwendungen komplett auf zentrale Server verlagert wird.

Die Referenz wurde in verschiedenen Produkten, beispielsweise der IBM Network Station oder der SUN JavaStation, realisiert. Der Fokus lag dabei darauf, Webanwendungen bzw. Java-Applets auszuführen. Während diesen Geräten – unter anderem mangels Unterstützung für die zeitgleich aufkommenden Remote-Protokolle zum Zugriff auf herkömmliche Anwendungen auf Basis der Microsoft Windows Plattform – kein wirtschaftlicher Erfolg beschieden war und sie sich somit nicht durchsetzen konnten, legten sie doch den Grundstein für eine Renaissance des Betriebsmodells der entfernten Präsentation.

So entwickelte SUN in der Nachfolge der JavaStation den „Sun Ray" Thin Client für den Remote-Zugriff auf Linux-Systeme sowie das hauseigene Betriebssystem Solaris. Auf der anderen Seite entstanden diverse Hardware- und Software-Lösungen für den Zugriff auf Microsoft Windows Server. So zielen Open Source Projekte, wie beispielsweise Thinstation[45] oder openThinClient[46] darauf ab, eine Software-Umgebung zu schaffen, die es ermöglicht, auf herkömmlicher PC-Hardware ein minimales Betriebssystem über das Netzwerk zu starten (vgl. [Knermann, 2007] und [Knermann, 2008b]). Dieses dient primär lediglich dem Zweck, Clientprogramme für gängige Remote-Protokolle und ggf. einen lokalen Webbrowser auszuführen, während die übrige Datenverarbeitung zentral auf Terminal und Web Servern erbracht wird.

[45] Thinstation: http://thinstation.github.io/thinstation/ (abgerufen am 01.02.2016).
[46] openThinClient – The Open Source Thin Client Solution: http://openthinclient.org (abgerufen am 01.02.2016).

Abbildung 2-26: Personal Computer (links) und Thin Client (rechts)

Quelle: [Hiebel, 2011]

Über das Netzwerk werden also – analog zum seriell angebundenen Terminal als historischem Vorbild – lediglich die Benutzereingaben sowie Video-/Audio-Ausgaben zwischen Client und Server übertragen. Diverse Hersteller, wie z. B. HP, IGEL Technology[47] oder Dell/Wyse[48], fertigen dedizierte Thin Client Hardware, die eben diesem Zweck dient. Entsprechenden Endgeräten ist gemein, dass es sich typischerweise um lüfterlose Geräte handelt, die mit auf Ein- und Ausgabe reduzierter Ausstattung deutlich weniger Leistung aufnehmen als gängige Personal Computer. Auf solchen Clients kommen durch die Anbieter angepasste Linux-Derivate, Windows Embedded Compact[49] oder Windows Embedded Standard[50] zum Einsatz.

Dienten die Thin Clients lange Zeit ausschließlich der entfernten Präsentation, sind die Anforderungen an die Client-Hardware – getrieben durch die Entwicklung der entsprechenden Remote-Protokolle – in jüngster Zeit wieder gestiegen, da fallweise Aufgaben vom Server auf den Client zurückverlagert werden. So überlässt Citrix, wie in Kapitel 2.1.3.4 erläutert, das Dekodieren diverser Multimedia-Formate einschließlich Adobe Flash dem Client, soweit dies die vom Benutzer wahrgenommene Geschwindigkeit der Sitzung verbessert. Mit der Version 7 von RDP hat Microsoft einen ähnlichen Weg eingeschlagen, um kontinuierliche Medien-Formate, wie z. B. WMA, WMV, MP3, DivX oder verschiedene MPEG-Dateien, auf Seiten des Clients zu

47 IGEL Technology: http://www.igel.com (abgerufen am 01.02.2016).

48 Dell/Wyse: http://de.wyse.com (abgerufen am 01.02.2016).

49 Bei Windows Embedded Compact handelt es sich um den Nachfolger des früheren Windows CE, eines separaten Entwicklungszweiges innerhalb der Microsoft Windows Familie. Da Windows CE eigenständige Gerätetreiber erfordert und herkömmliche Windows Applikationen darauf i. d. R. nicht ausgeführt werden können, somit auch in Falle von Clients gängiger Remote Protokolle Anpassungsbedarf besteht, kommt das Betriebssystem für Thin Clients heutzutage kaum mehr zum Einsatz.

50 Windows Embedded Standard (WES) ist eine Variante des regulären Windows Betriebssystems, die sich dadurch auszeichnet, dass sie in Komponenten-basierender Form ausgeführt ist. Dies bedeutet, dass Hersteller ein Betriebssystem mit genau den Modulen zusammenstellen können, die für den jeweiligen Anwendungsfall – beispielsweise Industriesteuerungen, Multifunktionsdrucker oder eben Thin Clients – notwendig sind. Ein solches modularisiertes System benötigt deutlich weniger Speicherplatz als eine vollwertige Windows Installation, findet somit auch auf einem Flash-Speicher Platz und ist dennoch binärkompatibel zu herkömmlichen Windows Treibern und Anwendungen.

dekodieren[51]. Bei diesen Ansätzen handelt es sich um Software-Lösungen, die neben generischer Hardware, d. h. geeigneter CPU, GPU sowie ausreichend dimensioniertem Hauptspeicher, lediglich einen Client für das jeweilige Remote-Protokoll sowie Codecs für die zu dekodierenden Medien erfordert.

Daneben haben Entwicklungen hin zu spezieller Hardware für Client und Server begonnen. Einer der Vorreiter in diesem Bereich war der Halbleiter-Entwickler Teradici mit dem Protokoll PC-over-IP (PCoIP), das ursprünglich als hardware-gestützte Remote-Lösung entworfen wurde. „Die Lösung besteht zum einen aus dem ‚TERA1200 Host Processor', der im Rechenzentrum zum Einsatz kommt, etwa in einem PC-Blade, einem Server-Blade oder beliebiger anderer Hardware im 19-Zoll Format oder Standgehäuse. Zum anderen wird ein Client mit dem ‚TERA1100 Portal Processor' benötigt, der mit dem Gegenpart auf der Seite des Servers kommuniziert. Gemeinsam implementieren die Prozessoren das Hardware-beschleunigte Protokoll ‚PC-over-IP (PCoIP)', das verspricht, PC-Leistungen über ein IP-Netzwerk zu liefern. Neben hochauflösender Grafik und HD-Audio soll die Anbindung nahezu beliebiger USB-Peripherie möglich sein. In bestehenden Systemen lässt sich der TERA1200 Host Processor mittels PCIe Erweiterungskarte nachrüsten. Weiterhin bieten bereits mehrere Hersteller OEM-Produkte, sowohl Blades als auch entsprechende Thin Clients, die ab Werk mit dem TERA-Chipsatz ausgestattet sind." (vgl. [Knermann, 2009a]).

Das Protokoll wurde im Jahr 2008 von VMware lizenziert. Anschließend entstand in einer Kooperation der beiden Firmen eine Software-Implementierung, die in der Desktop-Virtualisierungslösung VMware Horizon zum Einsatz kommt. Auch in diesem Fall weist die Entwicklung von Microsofts RDP in eine ähnliche Richtung, da Microsoft unter dem Label „RemoteFX" das Remote Desktop Protocol derart erweitert hat, dass Server-seitig vorhandene GPU zur Kompression grafischer Inhalte verwendet werden können. Um die so komprimierten Informationen auf dem Client wieder zu entpacken, kann an Stelle einer herkömmlichen CPU bzw. GPU auf dem Client optional auch ein dedizierter RemoteFX-Chip als anwendungsspezifische integrierte Schaltung (Application-Specific Integrated Circuit (ASIC)) zum Einsatz kommen[52].

Damit können im Bereich der Thin Clients zwei Kategorien von Geräten unterschieden werden. Auf der einen Seite finden sich Geräte, die eine möglichst große Menge an Remote-Protokollen unterstützen und so nicht auf einen konkreten Einsatzzweck festgelegt sind. Die Protokolle sind hierbei in der Regel in Software-Clients implementiert. Auf der anderen Seite sind Clients verfügbar, die für eine bestimmte Infrastruktur-Lösung optimiert sind, beispielweise als reiner Citrix XenDesktop/XenApp oder VMware Horizon Client. In diesem Fall sind die Protokolle in Hardware oder Software ausgeführt. Da die Endgeräte in letzterem Fall oftmals mit äußerst wenigen oder gar keinen lokalen Konfigurationsoptionen auskommen, ist für solche Clients auch der Begriff des „Zero Clients" gebräuchlich.

[51] http://blogs.msdn.com/b/rds/archive/2009/07/24/multimedia-redirection-improvements-in-windows-7-and-ws2008-r2-part-1.aspx (abgerufen am 01.02.2016).
[52] http://www.dabcc.com/article.aspx?id=16000&page=3 (abgerufen am 01.02.2016).

Somit ist auf Seiten des Clients über herkömmliche PC und Notebooks hinaus eine breite Auswahl an Hardware, speziell mit dem Fokus auf das Server Based Computing, gegeben. Die entsprechenden Entwicklungen im Bereich von Servern und der Infrastruktur im Rechenzentrum steht im Mittelpunkt des folgenden Abschnitts.

2.4.2 Server und Infrastruktur

Getrieben durch Virtualisierungstechnologien, die darauf abzielen, Ressourcen im Rechenzentrum zu konsolidieren, haben Bauformen und Hardwareeigenschaften von Serversystemen im Laufe der Zeit tiefgreifende Veränderungen erfahren. Ein Server definiert sich zunächst durch die auf ihm betriebene Software und die ihm zugedachten Aufgaben. So kann grundsätzlich jede Hardware vom herkömmlichen Personal Computer bis hin zum Großrechner als Server Verwendung finden, wenn sie entsprechende Dienste erbringt. Insbesondere in Bezug auf die x86 Architektur war es in der Vergangenheit gängige Praxis, viele dedizierte Systeme, auch als Stand-Alone Server bezeichnet, einzusetzen, die jeweils eine Betriebssystem-Instanz ausführten und nur einen oder wenige Dienste erbrachten. Sobald in einer Infrastruktur bestehend aus dedizierten Servern mehr Ressourcen nötig wurden, boten sich zwei Ansätze zur Erweiterung an (vgl. [Gai, 2010], S. 11 ff.):

- Im Falle der horizontalen Skalierung wurde die Anzahl an Servern erhöht und die Infrastruktur um weitere Systeme ergänzt. Dieser Ansatz wurde oftmals im Bereich der x86 Systeme gewählt. Ein Vorteil ist, dass so eine Infrastruktur aus günstig verfügbaren Standard-Komponenten aufgebaut werden kann und einzelne Systeme mit geringem Aufwand in und außer Betrieb genommen werden können. Nachteilig ist allerdings, dass eine steigende Menge einzelner Server einen nicht zu unterschätzenden Verwaltungsaufwand sowie weitere Betriebskosten[53] mit sich bringt.

- Demgegenüber sieht der Ansatz der vertikalen Skalierung vor, die Kapazität eines einzelnen Server-Systems zu erhöhen, indem dieses mit einer größeren Anzahl an Prozessoren, mehr Hauptspeicher und weiteren Eingabe-/Ausgabe-Geräten ausgestattet wird. Diese Herangehensweise wurde im Bereich der x86 Systeme vor allem durch das Aufkommen der Virtualisierungstechnologien befördert, da diese es möglich machten, entsprechend leistungsfähigere Hardware optimal auszulasten und dennoch weiterhin einzelne Dienste auf dedizierten, nun aber virtualisierten, Server-Instanzen zu betreiben.

Dem Prinzip der horizontalen Skalierung folgend hat insbesondere eine Entwicklung im Bereich der Server-Hardware seit Anfang des Jahrhunderts eine sehr starke Verbreitung erfahren, die sogenannten Blade Server oder kurz Blades. Dabei handelt es sich um modulare, physisch nicht selbständig funktionsfähige Server-Systeme, die in eine gemeinsame Betriebsumgebung, auch

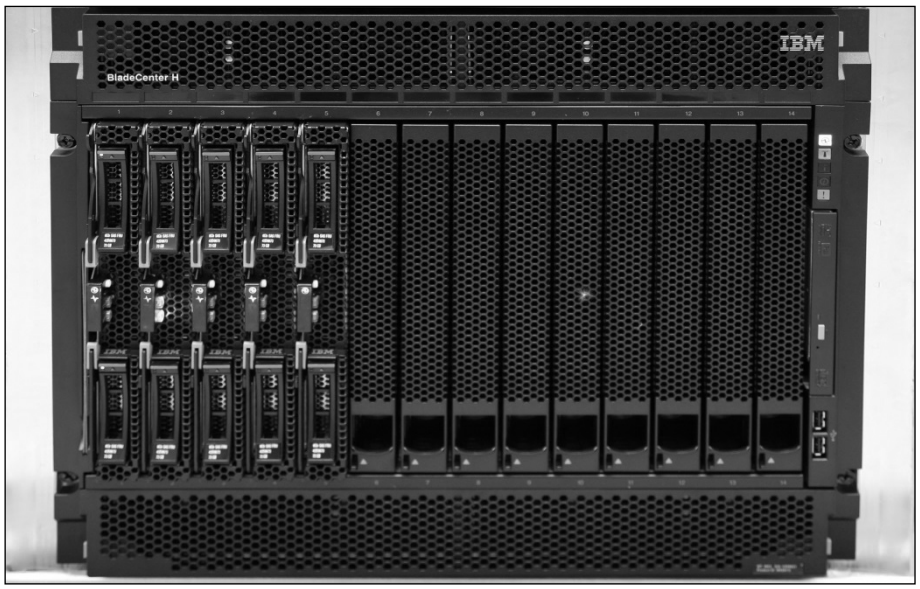
[53] Es ist gängige Praxis, einem Server lediglich eine oder möglichst wenige Aufgaben zuzuweisen, um das Konfliktpotenzial zwischen verschiedenen Softwarekomponenten auf einem einzelnen System zu minimieren und damit im Falle einer Störung dieses Servers nur wenige Dienste betroffen sind. Dies führte in der Vergangenheit zu einer rasanten Ausbreitung zahlreicher Systeme, die jeweils mit sehr geringer Last – typischerweise weniger als 5-10 Prozent CPU-Auslastung – laufen und somit überproportional Platz, Energie und Klimatisierung erfordern (vgl. [Gai, 2010], S. 15).

als Blade Chassis oder Blade Enclosure bezeichnet, eingeschoben werden können (Abbildung 2-27). Das Blade Chassis, welches in der Regel im 19-Zoll-Format ausgeführt ist und somit in herkömmlichen Server-Schränken montiert werden kann, bietet den Blades eine gemeinsame Stromversorgung, Lüftung, Netzwerkinfrastruktur sowie je nach Typ und Hersteller auch gemeinsamen Massenspeicher (vgl. [Watts, 2011]).

Jedes Blade stellt logisch eine unabhängige Recheneinheit mit eigenen Prozessoren, eigenem Hauptspeicher, Netzwerkkarten und je nach Typ lokalem Massenspeicher sowie ggf. zusätzlichen Erweiterungskarten dar. Charakteristisch für ein Blade ist allerdings, dass Netzwerk- sowie andere Peripheriegeräte nicht wie bei herkömmlichen Stand-Alone Servern über standardisierte Schnittstellen, wie beispielsweise eine RJ45-Buchse zum Anschluss eines Netzwerkkabels, direkt nach außen geführt werden.

Stattdessen wird ein Blade in einen passenden Schacht des Blade Chassis eingesteckt und über eine herstellerspezifische Schnittstelle mit einer zentralen Platine, der sogenannten Backplane oder auch Midplane, verbunden (Abbildung 2-28). Der Begriff der Midplane wird verwendet, da die Platine *in der Mitte* des Blade Chassis die Server auf der einen Seite mit den gemeinsam genutzten Komponenten – u. a. Netzteile, Lüfter, optische Laufwerke, USB- und serielle Schnittstellen, Netzwerk-Switches – auf der anderen Seite verbindet.

Abbildung 2-27: Blade Chassis, bestückt mit fünf Blade Servern



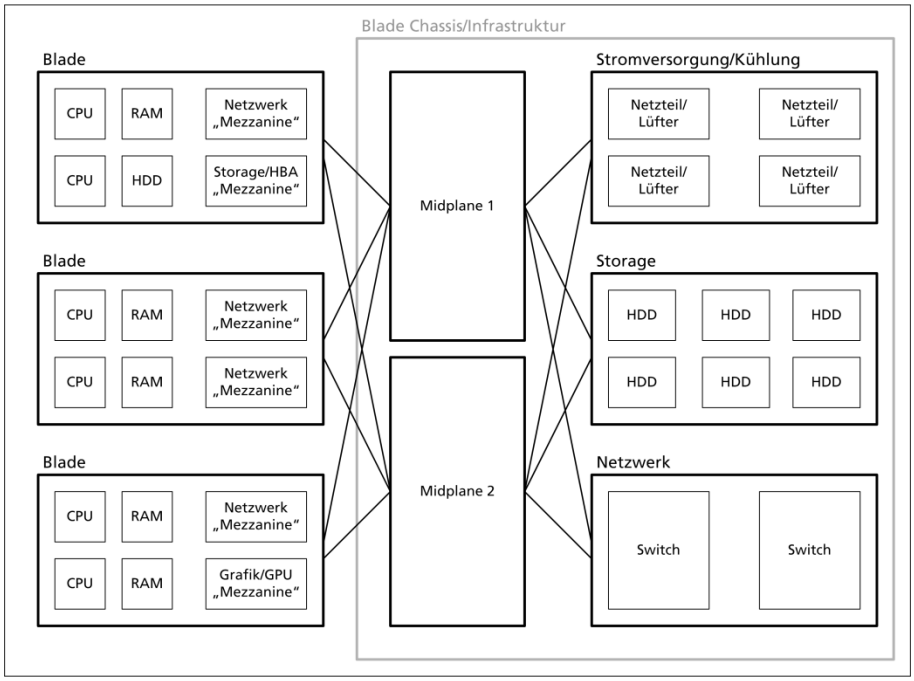
Quelle: [Hiebel, 2011]

Im Sinne höherer Verfügbarkeit können Midplane sowie die weiteren Komponenten des Blade Chassis redundant ausgelegt sein. Die modulare Architektur und die proprietären Formfaktoren bedingen es, dass in den Blade Servern keine herkömmlichen (PCI-) Erweiterungskarten zum Einsatz kommen können. Stattdessen finden sogenannte „Mezzanine“ Karten – nach dem englischen Begriff aus der Architektur für ein Halb- oder

Zwischengeschoss – Verwendung (vgl. [Gai, 2010], S. 15). Eine solche Karte ergänzt ein Blade beispielsweise um einen Netzwerk-Chipsatz oder einen dedizierten Host Bus Adapter (HBA) [54] zur Anbindung von Storage. Die jeweilige Funktionalität wird über die Midplane herausgeführt und entweder mit internen oder über standardisierte Schnittstellen mit externen Ressourcen verbunden.

Insbesondere der Anbindung von Massenspeicher kommt dabei in größeren Virtualisierungsinfrastrukturen hohe Bedeutung zu. So wird der lokale Massenspeicher[55] eines Blades oder ganz allgemein einer Server-Hardware typischerweise lediglich zur Installation des Hypervisors benutzt, während die Abbilder der VM mit ihren virtuellen Festplatten nicht lokal auf einem Server gespeichert werden, sondern in einem Storage-Bereich, der mehreren Servern gleichermaßen zugänglich ist. Dies ist Voraussetzung, um die erweiterten Funktionen aktueller Virtualisierungslösungen nutzen zu können. So wird es beispielsweise möglich, im Rahmen der sogenannten Live-Migration eine VM im laufenden Betrieb von einem physischen Host auf einen anderen zu verschieben.

Abbildung 2-28: Blade Architektur



Quelle: Eigene Darstellung

Zum Zweck optimaler Lastverteilung kann eine solche Migration auch durch die jeweilige Virtualisierungslösung automatisiert erfolgen. Weiterhin dient ein gemeinsamer zentraler

[54] Ein HBA ist mit einem separaten Prozessor ausgestattet, um die Zugriffe auf den Storage zu bearbeiten und so den Prozessor des Hosts zu entlasten.

[55] Dabei kann es sich um eine oder mehrere Festplatten, fallweise aber auch lediglich um einen Flash-Speicher, beispielsweise einen USB-Stick, handeln. Der lokale Speicher wird auch als Direct Attached Storage (DAS) bezeichnet.

Speicher der Erhöhung der Verfügbarkeit, indem beim Ausfall eines Hosts die betroffenen VM automatisiert auf einem anderen Host neu gestartet werden können. Hierbei sind der blockorientierte Zugriff auf ein Storage Area Network (SAN) sowie der dateiorientierte Zugriff auf einen Network Attached Storage (NAS) zu unterscheiden. In ersterem Fall „wird der externe Speicher (…) dem Client (in unserem Falle (…) dem Host) wie eine leere Festplatte präsentiert. Der Client muss sich um die Verwaltung des Dateisystems dieser Festplatte selbst kümmern." (vgl. [Ahnert, 2007], S. 56). Für einen Hypervisor auf einem physischen Host bedeutet dies, die als Logical Unit Number (LUN) adressierte logische Festplatte im SAN mit einem ihm eigenen Dateisystem zu formatieren, so dass mehrere Hosts gleichzeitig auf diesen Speicher zugreifen können. Physisch erfolgt der Zugriff auf den Storage wahlweise über einen speziellen HBA zur Anbindung von Storage per Fibre Channel oder iSCSI, wobei in letzterem Fall die Anbindung auch über eine herkömmliche Ethernet-Netzwerkkarte realisiert werden kann. Auch der dateibasierte Zugriff auf einen NAS, also eine Dateifreigabe im Netzwerk, ist im Kontext der bislang diskutierten Virtualisierungslösungen relevant. So sind beispielsweise die in Kapitel 2.3 erwähnten Citrix Provisioning Services in der Lage, eine gemeinsame CIFS[56]-Freigabe als Store für die Ablage der Images virtueller Maschinen zu verwenden.

Im Zusammenhang mit den zuvor erwähnten Blades kann eine Storage-Einheit im Blade Chassis als SAN/NAS Verwendung finden, um einen gemeinsamen Speicherbereich für mehrere Blades zu schaffen (Abbildung 2-28) und so die Verfügbarkeit über mehrere Blades zu erhöhen. Weiterhin kann ein externes SAN-/NAS-System zum Einsatz kommen, um eine Virtualisierungsinfrastruktur über mehrere ggf. räumlich getrennte Blade Chassis zu etablieren.

2.5 Provider-Modelle

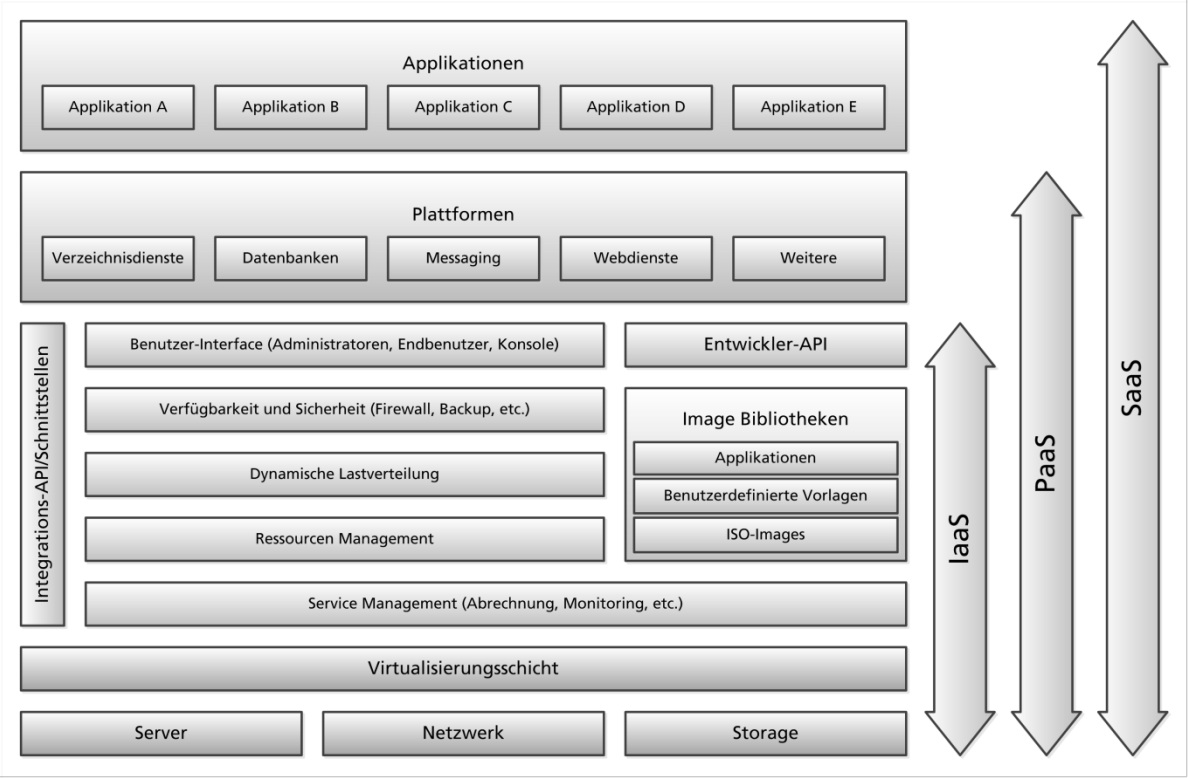
Neben der internen Nutzung der zuvor erläuterten Technologien innerhalb eines Unternehmens haben sich unter dem Stichwort des Cloud Computings zunehmend Provider-Modelle etabliert, in deren Kontext externe Service-Provider Virtualisierungstechnologien nutzen, um anderen Unternehmen IT-Ressourcen und -Dienste anzubieten. Zur Differenzierung entsprechender Angebote werden typischerweise drei Betriebsmodelle unterschieden, nämlich „Infrastructure as a Service (IaaS)", „Platform as a Service (PaaS)" und „Software as a Service (SaaS)" (vgl. [BITKOM, 2009], S. 22 ff.). Diese Modelle bauen aufeinander auf und haben zum Ziel, Kunden nach Bedarf in flexibler Weise zusätzliche Verarbeitungskapazitäten zur Verfügung zu stellen (vgl. Abbildung 2-29).

- **Infrastructure as a Service (IaaS):** In diesem Fall „nutzt ein Kunde Server, Storage, Netzwerk und die übrige Rechenzentrums-Infrastruktur als abstrakten, virtualisierten Service über das Internet. Diese Services werden typischerweise nutzungsabhängig abgerechnet…" (vgl. [BITKOM, 2009], S. 24). Dies bedeutet, dass die Kunden ihre eigene Rechenzentrumsinfrastruktur flexibel erweitern können und bedarfsorientiert

[56] Das Common Internet File System (CIFS) findet unter Betriebssystemen der Microsoft Windows Familie typischerweise Verwendung für die Bereitstellung von Dateifreigaben im Netzwerk.

zusätzliche Ressourcen nur dann abrufen, wenn diese tatsächlich benötigt werden. Die Investition in eigene Hardware entfällt somit. Beispiele für entsprechende Angebote sind der Amazon Simple Storage Service (Amazon S3)[57] speziell für die Anbindung von cloud-basiertem Datenspeicher oder die Amazon Elastic Compute Cloud (Amazon EC2)[58] als Umgebung zur Ausführung von virtuellen Maschinen mit unterschiedlichen Betriebssystemen.

Abbildung 2-29: Ausprägungen verschiedener Provider-Modelle



Quelle: Eigene Darstellung, nach Mach Technology Group, 2011[59]

- **Platform as a Service (PaaS):** Dieses Betriebsmodell erweitert das IaaS-Konzept um eine weitere Schicht. Aufbauend auf IaaS erhalten Kunden zudem eine Laufzeitumgebung als „Anwendungs-Infrastruktur in Form von technischen Frameworks (Datenbanken und Middleware) oder die gesamte Anwendungssoftware" (vgl. [BITKOM, 2009], S. 26). Charakteristisch hierfür ist, dass die Kunden eine Plattform mit einem definierten Application Programming Interface (API) bzw. Schnittstellen zu anderen Webservices erhalten, mit denen sie angepasste (Web-) Applikationen entwickeln können. Ausprägungen entsprechender Angebote sind

[57] Amazon Simple Storage Service (Amazon S3): http://aws.amazon.com/de/s3/ (abgerufen am 01.02.2016).
[58] Amazon Elastic Compute Cloud (Amazon EC2): http://aws.amazon.com/de/ec2/ (abgerufen am 01.02.2016).
[59] Präsentation „Integrated Managed Services – ‚Next Generation' of full-service ICT solution outsourcing": http://www.slideshare.net/openstackcommgr/mach-technology (abgerufen am 01.02.2016)

beispielsweise die Google App Engine[60], Microsoft Windows Azure[61] oder auch Jelastic[62] für Java-Anwendungen.

- **Software as a Service (SaaS):** Hierbei werden die zuvor erläuterten Betriebsmodelle um eine weitere Schicht ergänzt. Im Rahmen von SaaS-Angeboten erhalten die Kunden komplette Standardsoftware. Dabei kann es sich zum einen um speziell im Hinblick auf diesen Einsatzzweck entwickelte Software und Webdienste handeln, z. B. Google Docs[63], die Webanwendungen von Microsoft Office 365[64] oder Salesforce.com[65]. Zum anderen können die in den Kapiteln 2.1 und 2.2 vorgestellten SBC Technologien Anwendung finden, um herkömmliche Windows-Anwendungen im Rahmen von SaaS anzubieten, ohne die Anwendungen selbst anpassen zu müssen.

Wie zuvor erläutert, können mittels Server Based Computing sowohl einzelne Anwendungen als auch komplette Desktop-Umgebungen per entfernter Präsentation veröffentlicht werden. Für die letztere Variante als Sonderfall von SaaS hat die inzwischen in VMware aufgegangene Firma Desktone den Begriff „Desktop as a Service (DaaS)" geprägt[66]. Dieser findet in der Literatur jedoch nicht durchgängig Verwendung und in technischer Hinsicht besteht kein Unterschied zu SaaS.

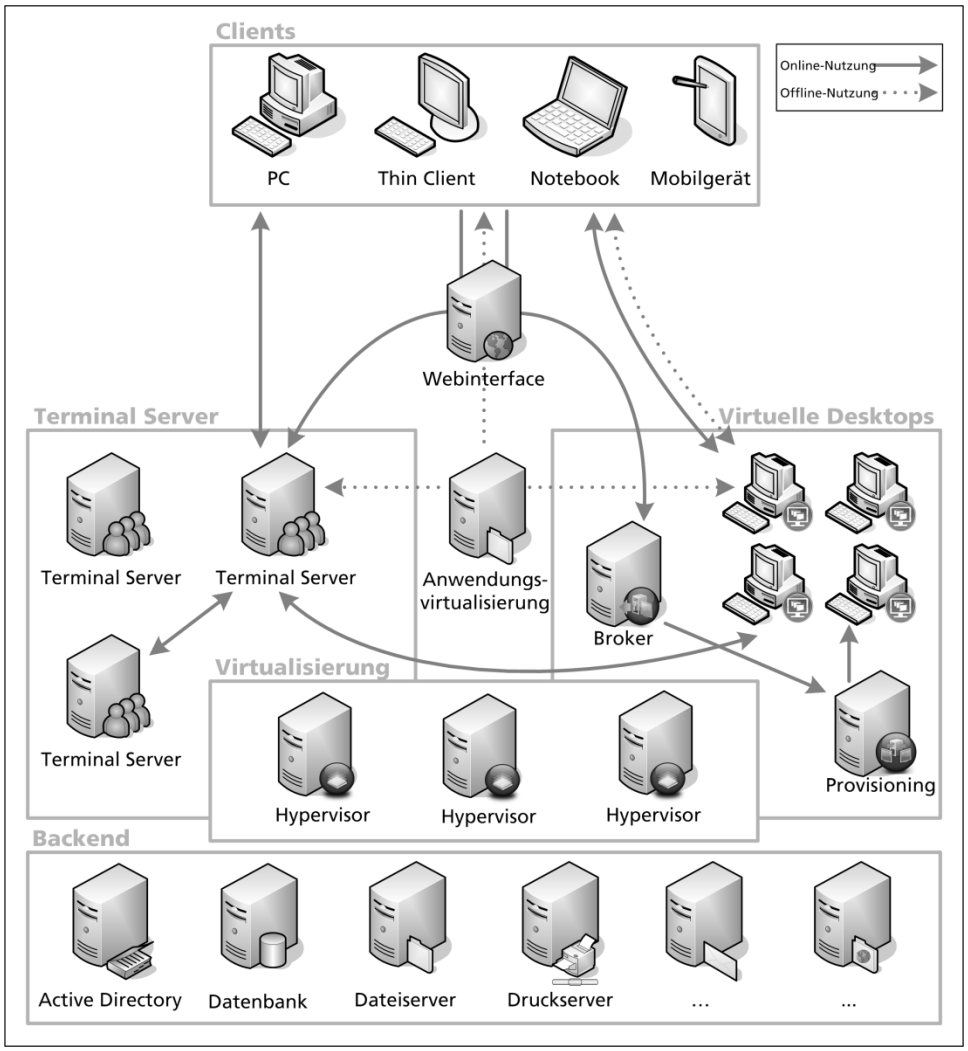
2.6 Zusammenfassung

Die in diesem Kapitel vorgestellten Technologien bilden die Basis für die verschiedenen Ausprägungen des Server Based Computings und lassen sich – ausgehend von der Prämisse, dass primär auf Microsoft Windows Betriebssysteme aufsetzende Anwendungen bereitzustellen sind – zu folgender Gesamtsicht einer SBC Infrastruktur zusammenfügen (Abbildung 2-30). Die Basis bilden grundlegende Dienste im Backend, insbesondere mindestens eine Active Directory Domäne als Verwaltungseinheit für sämtliche Computer- und Benutzerobjekte und somit mindestens eine (virtuelle) Maschine als Domain Controller, welcher die Active Directory-Dienste ausführt. Hinzu kommen mindestens ein Datenbankserver, der die Konfiguration der SBC Infrastruktur aufnimmt, sowie weitere Dienste, soweit diese von den Terminal Servern und virtuellen Desktops benötigt werden. Dabei kann es sich beispielsweise um Datei- und Druckserver handeln. Auch diese Backend-Server werden heutzutage typischerweise als virtuelle Maschinen realisiert. Die Virtualisierungsschicht bildet aber vor allem die Basis für die Bereitstellung der Terminal Server und virtuellen Desktops

[60] Google App Engine: https://developers.google.com/appengine/ (abgerufen am 01.02.2016).
[61] Microsoft Windows Azure: http://www.windowsazure.com/de-de/ (abgerufen am 01.02.2016).
[62] Jelastic: http://jelastic.com (abgerufen am 01.02.2016).
[63] Google Docs: http://www.google.com/apps/intl/de/business/docs.html (abgerufen am 01.02.2016).
[64] Microsoft Office 365: www.microsoft.com/Office365 (abgerufen am 01.02.2016).
[65] Salesforce.com CRM Software & Online CRM System: https://www.salesforce.com/de/ (abgerufen am 01.02.2016).
[66] „VMware grabs the company who trademarked "desktop as a service"": http://www.citeworld.com/article/2115160/cloud-computing/why-vmware-bought-desktone.html (abgerufen am 01.02.2016).

selbst[67]. Auf einem oder mehreren Hosts, die jeweils einen Typ 1 Hypervisor ausführen, welcher in der Regel hardware-gestützte Virtualisierung in Verbindung mit binärere Translation oder Paravirtualisierung verwendet, werden dazu entsprechende logische Verwaltungseinheiten für Server- und Desktop-VM abgebildet.

Abbildung 2-30: SBC Infrastruktur, Gesamtsicht



Quelle: Eigene Darstellung

Eine Site virtueller Desktops beinhaltet einen oder mehrere Broker, die zwischen physischen Client-Computern und den Desktop-VM vermitteln. Letztere werden nach Bedarf von einem oder mehreren Provisioning Servern, die mit der Virtualisierungsschicht, dem Storage-

[67] Je nach Leistungsanforderungen können natürlich Terminal Server wie auch virtuelle Desktops fallweise ohne Virtualisierungsschicht direkt auf einer Hardware installiert werden. Um die Darstellung der Gesamtsicht nicht zu komplex geraten zu lassen, wird dieser Fall aber in der Abbildung der Gesamtsicht nicht weiter berücksichtigt.

Subsystem und dem Active Directory interagieren, erzeugt, (neu)gestartet, heruntergefahren[68]. Die Desktop-VM veröffentlichen mit den vorgestellten Methoden und Protokollen der entfernten Präsentation wahlweise eine vollständige Desktop-Umgebung oder einzelne Applikationen an die Clients. Um das Image, aus dem die Desktops provisioniert werden, möglichst generisch zu halten, können die Desktops ihrerseits Anwendungen mittels der entfernten Präsentation von Terminal Servern online starten oder als mit den Methoden der Anwendungsvirtualisierung gekapselte Pakete offline hinzuladen.

Auch eine Farm von Terminal Servern veröffentlicht im Rahmen der entfernten Präsentation entweder komplette Desktops oder einzelne Anwendungen zur Online-Nutzung an ihre Clients. Je nach Verteilung der Applikationen über die Server der Farm können auch Terminal Server ihrerseits weitere Applikationen von anderen Terminal Servern oder gar virtuellen Desktops starten oder als virtualisiertes Paket beziehen. So ist es beispielsweise denkbar, mehrere Terminal Server mit einer standardisierten Arbeitsumgebung für alle Benutzer, z. B. einem Office Paket, bereitzustellen und selten bzw. nur von wenigen Benutzern benötigte Spezialanwendungen fallweise auf einem weiteren Terminal Server auszuführen.

Eine Migration beispielsweise hin zum in Kapitel 2.3.2 erläuterten Citrix XenDesktop 7.x erlaubt es, auf Ebene der Terminal Server und virtuellen Desktops die Komplexität zu reduzieren, da beide Konzepte zu einer integrierten Betriebsumgebung zusammengeführt werden können (vgl. Abbildung 2-23). Diese nutzt eine Verwaltungskonsole, eine Provisionierungstechnik und eine Datenbank, um Mehrbenutzer-Systeme auf Basis von Terminal Servern und gleichermaßen dedizierte virtuelle Desktops auf Basis von Client-Betriebssystemen bereitzustellen.

Als Clients einer solchen Infrastruktur kommen sämtliche Endgeräte in Frage, die in der Lage sind, ein Clientprogramm für das zur entfernten Präsentation verwendete Protokoll und ggf. einen Webbrowser auszuführen. Neben herkömmlichen Fat Clients – Desktop-PC oder Notebooks – kann sich dabei um Thin Clients, Netbooks, Smartphones oder Tablets handeln. Clientprogramm oder Webbrowser kontaktieren einen Webserver oder anderweitigen zentralen Broker, der seinerseits die für den anfragenden Benutzer freigegebenen Ressourcen ermittelt und aggregiert. Aus der Menge der verfügbaren Ressourcen kann dann die gewünschte Anwendung ausgewählt werden, woraufhin eine direkte Verbindung zu einem Terminal Server oder virtuellen Desktop aufgebaut wird.

2.7 Abgrenzung

Wie aus der Gesamtsicht einer SBC Infrastruktur (Abbildung 2-30) ersichtlich, besteht technisch die Möglichkeit, einzelne Anwendungen oder vollständige Desktops zur Offline-Nutzung auf einen Client zu übertragen. Insbesondere für mobile Endgeräte eröffnet dies die Option,

[68] Der Ansatz, viele VM automatisiert mittels Provisionieren zu erzeugen und zu verwalten, bietet sich gleichermaßen natürlich auch für das Management der Terminal Server an. Auch dies wurde in der Abbildung aus Gründen der Vereinfachung nicht dargestellt, sollte aber bei der Planung und praktischen Umsetzung einer SBC Infrastruktur nicht unberücksichtigt bleiben.

Ressourcen auch ohne dauerhafte Netzwerkverbindung zur SBC Infrastruktur zu nutzen. In beiden Fällen sind allerdings nach heutigem Stand der Technik Einschränkungen zu berücksichtigen. So erfordert der Anwendungsfall einzelner virtualisierter Applikationen, dass diese auf dem Endgerät eine passende Laufzeitumgebung vorfinden. Dies beschränkt die Anwendbarkeit in der Regel auf Clients, die über ein Betriebssystem der Microsoft Windows Familie und genügend Leistung sowie Speicherplatz verfügen, um die Anwendung vollständig zu laden und auszuführen.

Zur Offline-Nutzung kompletter Desktops muss der Client dagegen einen Typ 1 oder Typ 2 Hypervisor ausführen. Es existieren für beide Ausprägungen mehrere Lösungen, die dies realisieren, im Fall des Typ 1 Hypervisors bieten die heute verfügbaren Alternativen jedoch keine direkte Integration mit der Infrastruktur zur Online-Bereitstellung von Desktops. Zwar verfügt die in Kapitel 2.3.4 beispielhaft erwähnte Lösung XenClient über eine Management-Komponente, über die Images zentral bereitgestellt und auch von Clients zurück in das Rechenzentrum übertragen werden können.

Die gesicherten Images können aber nicht ohne weiteres zentral gestartet und im Rahmen der entfernten Präsentation genutzt werden, so dass die genannten Lösungen eher dem Bereich des Clientmanagements denn des Server Based Computings zuzurechnen sind (vgl. [Knermann, 2014]). Weiterhin erfordert die Offline-Nutzung natürlich grundlegend, dass überhaupt ein Hypervisor auf dem Endgerät installiert ist, im Falle des Typ 1 Hypervisors sogar *an Stelle* eines herkömmlichen Betriebssystems.

Die folgenden Untersuchungen werden den Fokus auf die Nutzung von Technologien des Server Based Computings zur Veröffentlichung von Anwendungen und Desktopumgebungen an Unternehmens-interne Kunden legen. Ziel ist es dabei, möglichst viele Kunden zu bedienen und dies möglichst unabhängig davon, welche Art von Endgerät und welches Betriebssystem ein einzelner Kunde einsetzt. Auf Grund der zuvor genannten technischen Einschränkungen wird der Anwendungsfall der Offline-Nutzung im Folgenden nicht weiter berücksichtigt und es wird der Online-Zugriff auf Anwendungen und Desktops mittels entfernter Präsentation im Mittelpunkt der Betrachtungen stehen.

3 Sicherer Zugriff auf SBC Infrastrukturen

Neben der vom Benutzer wahrgenommenen Geschwindigkeit eines Systems dürfen natürlich auch die Belange der IT-Sicherheit nicht unberücksichtigt bleiben. Insbesondere in SBC Anwendungsszenarien ist den Endanwendern der sichere Zugriff auf zentral bereitgestellte Applikationen auch von außerhalb eines geschlossenen Unternehmensnetzes zu ermöglichen. Beim Zugriff von anderen Standorten aus, und auch allgemein im Falle mobiler Nutzung, soll dabei möglichst die gleiche Qualität und Geschwindigkeit geboten sein wie bei einer Nutzung vor Ort. Dabei dürfen externen Nutzern ausschließlich die ihnen zugedachten Applikationen zur Verfügung stehen. Die Sicherheitsbestrebungen dürfen jedoch nicht zu Lasten der Benutzbarkeit gehen, da der Erfolg von SBC Projekten in erheblichem Maße von der Akzeptanz der Endanwender abhängt.

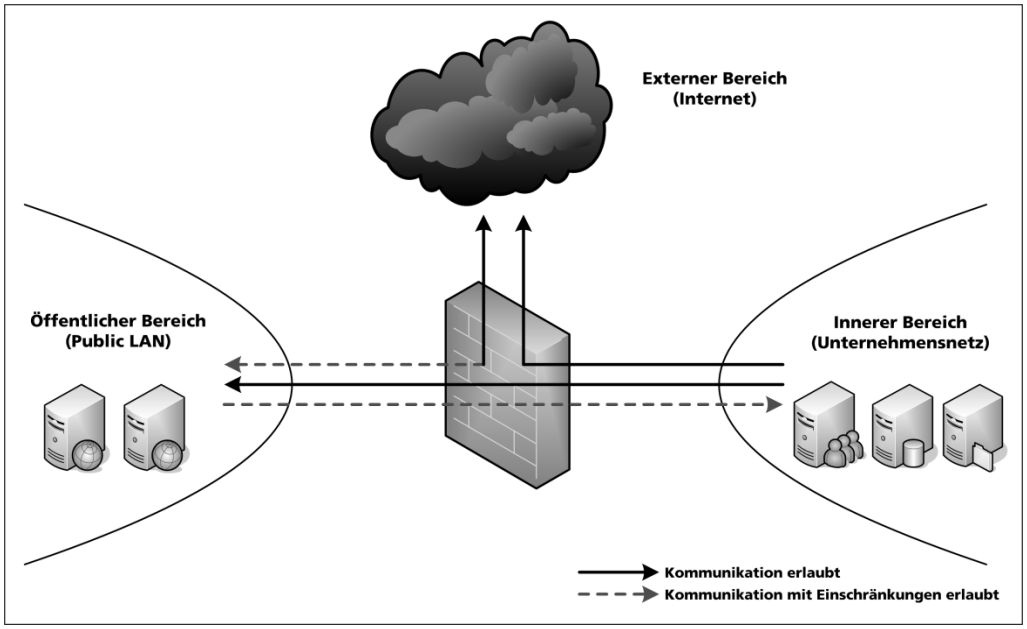
3.1 Grundlegende Netzwerkarchitektur

Im fortlaufenden Prozess der Etablierung und Pflege von Sicherheitsrichtlinien hat sich ein Zonenmodell als gängige Praxis beim Aufbau einer Netzwerkarchitektur etabliert (Abbildung 2-1). Eine solche Architektur wird nach dem Prinzip, alle Verbindungen auf Netzwerkebene zu verbieten, die nicht explizit erlaubt sind, in drei Bereiche mit ansteigendem Sicherheitslevel unterteilt – den externen Bereich (Internet), den öffentlichen Bereich (Public LAN), auch als Demilitarisierte Zone (DMZ) bezeichnet sowie den inneren Bereich, also das eigentliche private LAN des Unternehmens.

Eine paketfilternde Firewall mit „Stateful Inspection“ (vgl. Kapitel 3.2.1) als zentrale Komponente überwacht, dass ausschließlich Zugriffe aus Bereichen mit einem höheren Sicherheitslevel auf solche mit einem niedrigeren Sicherheitslevel möglich sind. Für spezielle Dienste, wie z. B. E-Mail-, HTTP- oder FTP-Verkehr, also allgemein Dienste, die einen Zugriff von außen, d. h. von niedrigeren in höhere Sicherheitsbereiche erfordern, wird keine direkte Verbindung erlaubt. In einem solchen Fall ist immer ein Gateway (vgl. Kapitel 3.2.2 und 3.2.3) einzusetzen. Unter Beachtung der zu erfüllenden Sicherheitsanforderungen (vgl. [Dannbacher, 2002], Seite 5) Authentifizierung und Autorisierung[69], Anonymität[70], Verbindlichkeit, Vertraulichkeit und Integrität[71], Verfügbarkeit und Konsistenz[72] werden im Folgenden die unterschiedlichen Konzepte zur Absicherung der verschiedenen Zonen erläutert.

[69] Zugriff nur nach Beweis der Identität und nur auf die Ressourcen, für die der Benutzer explizit befugt ist.
[70] Im Sinne des Schutzes personenbezogener Daten vor Leistungs- und Verhaltenskontrolle.
[71] Zwischen einem definierten Absender und einem definierten Empfänger werden Daten geschützt vor unbefugtem Zugriff und Manipulation ausgetauscht.
[72] Das System muss kontinuierlich bereitstehen und sich den Benutzern gegenüber erwartungskonform verhalten.

Abbildung 3-1: Grundlegende Netzwerkarchitektur



Quelle: Eigene Darstellung

3.2 Sicherer Zugriff über unsichere Netze

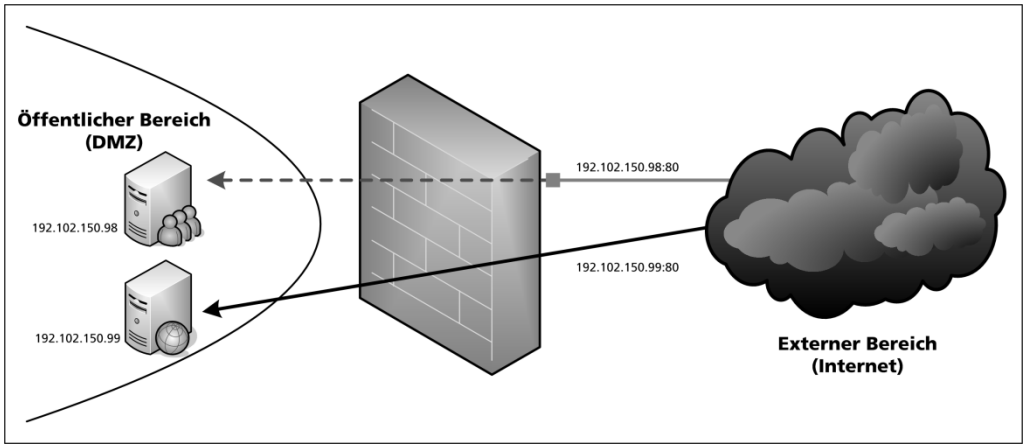
Bereits im Basiswerk zu Firewall-Technologien für den Einsatz von Internet-Anwendungen (vgl. [Cheswick, 1994], Seite 51 ff.) werden die drei Ausprägungen Paketfilter, Circuit Level Gateway sowie Application-Level Gateway unterschieden, deren grundlegende Funktionsprinzipien bis heute Bestand haben.

3.2.1 Paketfilter

Als Paketfilter kommt ein Router auf der Netzwerkschicht des ISO/OSI-Referenzmodells (vgl. [Hunt, 1992], Seite 5 ff.) zum Einsatz, dessen Funktionsprinzip darin besteht, basierend auf den Informationen aus IP-Adressen, Ports und der Kommunikationsrichtung Datenpakete passieren zu lassen oder nicht. Über das Sicherheitskonzept der „Stateful Inspection" wird dabei bidirektionale Kommunikation möglich, ohne dass aufwändige Filter in beiden Richtungen zu definieren sind. Eine solche Firewall kann eine tiefere Analyse der eintreffenden Datenpakete vornehmen – ggf. bis zu Informationen aus der Applikationsebene – und so Pakete über die Informationen der Netzwerkebene hinaus verbindungsorientiert zuordnen.

Pakete, die aus einem niedrigeren Bereich als Antwort auf eine von einem höheren Bereich initiierte Verbindung verschickt werden, können passieren, während Pakete zurückgewiesen werden, wenn die Verbindung ihren Ursprung in einem niedrigeren Bereich hat. Mittels dieser Technik lässt sich der Grundschutz der DMZ realisieren. So ist es beispielsweise möglich, HTTP-Zugriffe über den TCP-Port 80 aus dem Internet ausschließlich auf einen Web-Server zu erlauben, auf alle anderen Server aber zu verbieten (Abbildung 3-2).

Abbildung 3-2: Paketfilternder Router



Quelle: Eigene Darstellung

Dieser Ansatz allein reicht jedoch nicht aus, die Sicherheitsanforderungen bei der Bereitstellung von Anwendungen mittels SBC zu erfüllen, da lediglich auf Basis von Herkunftsadresse und Port eine externe Anfrage nicht sicher authentifiziert werden kann. Auch können grundsätzlich bei der Kommunikation über öffentliche Netze Kriterien wie Integrität oder Verbindlichkeit nicht zugesichert werden.

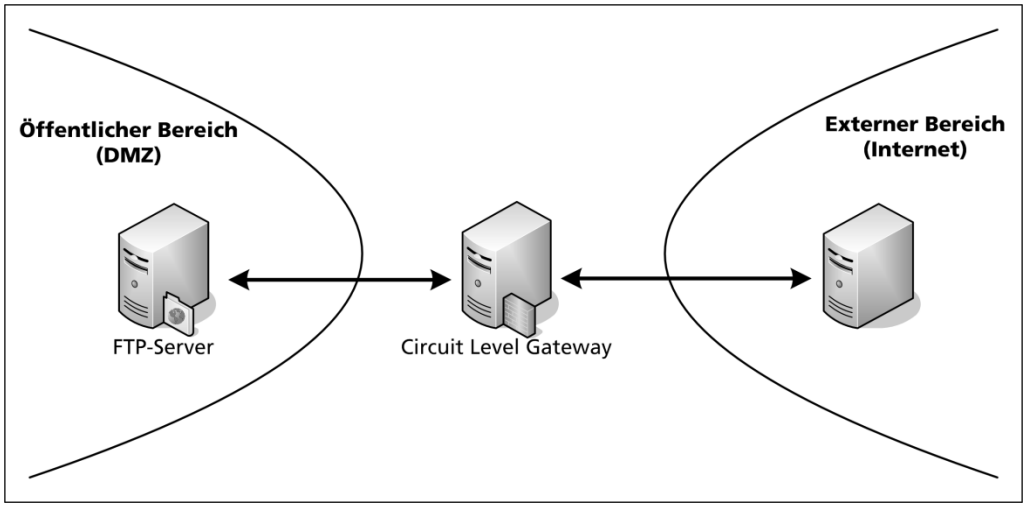
Dementsprechend dürfen über eine solche Verbindung keine geschäftskritischen Daten ausgetauscht werden, was vor allem bedeutet, dass zwischen dem externen und dem inneren Bereich keine direkte Kommunikation zustande kommen darf. Dies wird technisch nicht nur dadurch verhindert, dass keine entsprechenden Regeln auf dem paketfilternden Router definiert werden, sondern auch durch Verwendung privater IP-Adressen, die im Internet nicht bekannt sind und daher nicht direkt kontaktiert werden können. Hier erfordert schon die Notwendigkeit der Umsetzung von privaten auf offizielle IP-Adressen den Einsatz weiterer Technologien.

Für externe Nutzer müssten entsprechende Terminal Server im öffentlichen Bereich des Netzes installiert werden. Für einen reinen Microsoft Terminal Server müsste dann der TCP-Port 3389 für RDP geöffnet werden, für Terminal Server mit der Systemerweiterung Citrix XenApp die TCP-Ports 1494 (ICA) bzw. 2598 (CGP). Aber auch dies stellt eine Verletzung der Sicherheitsanforderungen dar, da hiermit (zumindest auf Protokollebene) Authentifizierung und Integrität der Kommunikation nicht sichergestellt werden können. Ein weiteres Problem erwächst aus der ebenfalls geforderten Benutzbarkeit, die darunter leidet, dass viele externe Nutzer ebenso aus gesicherten Unternehmensnetzwerken heraus zugreifen, deren Aufbau der in Kapitel 3.1 dargestellten Architektur entspricht. In der Regel ist es nicht vorgesehen, aus einem privaten Netzbereich heraus direkte Verbindungen nach extern zu initiieren, geschweige denn für Protokolle über Standards wie HTTP, HTTPS oder FTP hinaus. Auch hier sind also weitergehende Technologien gefragt, um SBC Infrastrukturen erfolgreich etablieren zu können.

3.2.2 Circuit Level Gateway

Ein Gateway – oftmals auch als „Bastion Host“ bezeichnet – leistet kein transparentes Routing auf Netzwerkebene. Vielmehr handelt es sich um ein System, das mit zwei oder mehreren Netzwerkbereichen über dedizierte Anschlüsse verbunden ist und zwischen diesen als Vermittler (Relay) agiert (Abbildung 3-3). Ein Circuit Level Gateway, wie es z. B. für FTP- oder HTTP-Angebote zum Einsatz kommen kann, arbeitet auf der Transportschicht des ISO/OSI-Referenzmodells.

Abbildung 3-3: Circuit Level Gateway



Quelle: Eigene Darstellung

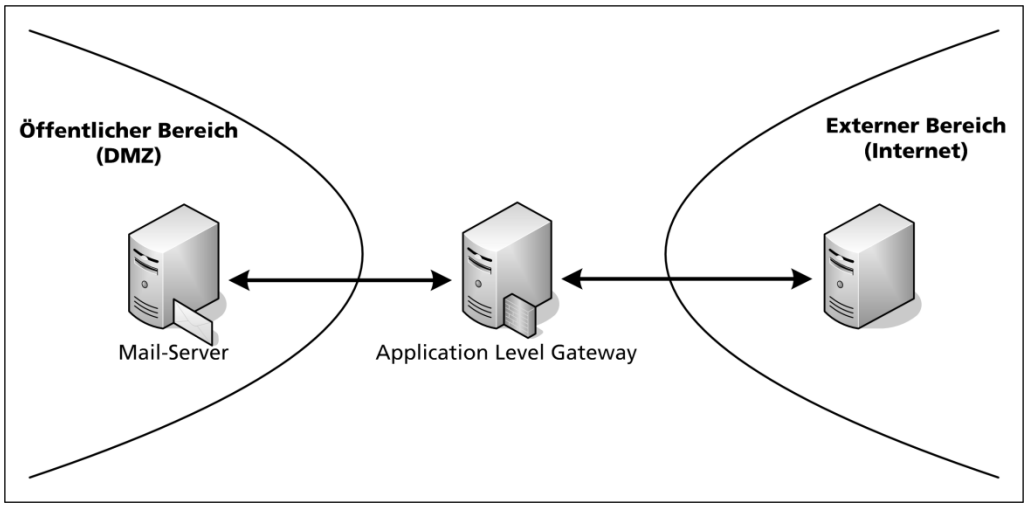
Nach einer Authentifizierung am Gateway stellt dieses verbindungsorientiert eine transparente Kommunikation zwischen zwei Partnern bereit, so dass der eigentliche Datenfluss sich der Kontrolle des Gateways entzieht. Da das Circuit Level Gateway keine portbasierte Filterung vornimmt, wird es im praktischen Einsatz in der Regel mit anderen Sicherheitslösungen kombiniert.

3.2.3 Application-Level Gateway

Ein Application-Level Gateway arbeitet auf der Applikationsschicht und somit der obersten Ebene des ISO/OSI-Referenzmodells. Im Gegensatz zum Circuit Level Gateway, das pauschal verbindungsorientierte Kommunikationsbeziehungen aufbaut und somit beliebige Dienste unterstützen kann, lässt ein Application-Level Gateway nur eine beschränkte Anzahl von Diensten zu, da für jeden zu nutzenden Dienst ein separater Vermittlungs-Prozess als Proxy zu betreiben ist (vgl. [Martius, 2000], S. 58). Aufgabe eines solchen Proxys ist nicht allein die Weiterleitung des Datenstroms einer bestimmten Applikation. Vielmehr kann der Proxy den Datenstrom interpretieren, kontrollieren und gegebenenfalls auf Applikationsebene in die Kommunikation eingreifen. Ein Beispiel hierfür ist der E-Mail-Verkehr (vgl. Abbildung 3-4) über das Simple Mail Transfer Protocol (SMTP), bei dem ein Gateway in der DMZ die gesamte E-Mail-Kommunikation für eine Organisation abwickelt. Dabei verbirgt das Gateway die interne

Komplexität der Organisation, indem der Außenwelt lediglich das Gateway bekannt sein muss, an das E-Mail-Nachrichten übermittelt werden. Mittels der Informationen aus dem E-Mail-Header – also Informationen der Applikationsebene – trifft das Gateway die Entscheidung, wie mit der Nachricht weiter zu verfahren ist.

Abbildung 3-4: Application-Level Gateway



Quelle: Eigene Darstellung

So kann z. B. die Nachricht an einen internen E-Mail-Server weitergeleitet oder auf Basis einer Liste der bekannten Benutzer innerhalb der Organisation bereits auf dem Gateway als unzustellbar abgewiesen werden. Des Weiteren lässt sich das Gateway als inhaltsbasierender Filter einsetzen, um beispielsweise unerwünschte Spam- oder Viren-Mails in Quarantäne zu nehmen.

Als weiteres Beispiel sei ein Web-Proxy, wie z. B. der verbreitete Squid[73], angeführt, der Rechnern aus dem inneren Bereich erlaubt, per HTTP mit dem Internet zu kommunizieren. Anstatt direkt einen externen Web-Server anzusprechen, stellt ein Rechner dabei die Anfrage an den Squid, der wiederum mit dem externen Host kommuniziert und zwischen den Kommunikationspartnern vermittelt. Dabei agiert auch dieser Proxy nicht transparent, sondern nutzt Informationen der Applikationsebene, um z. B. in seiner Funktion als Cache Seiten zwischen zu speichern und beim nächsten Aufruf nicht erneut aus dem Internet anzufordern, sondern die lokale Kopie an den anfragenden Client zurückzugeben.

Bezogen auf das Ziel, den sicheren Zugang zu Terminal Servern zu etablieren, zeigen die Beispiele auf, dass das Prinzip des Application-Level Gateways nicht isoliert betrachtet werden kann. Da es auf der obersten Ebene des ISO/OSI-Referenzmodells zum Einsatz kommt, ist offensichtlich, dass das Protokoll der Applikationsebene bzw. *die Applikation selbst* von der

[73] http://www.squid-cache.org (abgerufen am 01.02.2016).

Existenz des Gateways wissen und mit diesem umgehen muss. Im ersten Beispiel ist dies tief im SMTP-Protkoll, den Client- und Server-Komponenten verankert, die z. B. Routinginformationen aus speziellen DNS-Einträgen (vgl. [Cheswick, 1994], S. 75/76) beziehen. Im zweiten Beispiel muss der Client-Komponente, dem Web-Browser, die Adresse des Proxy-Servers bekannt gegeben werden, so dass dieser seine Anfragen an den Proxy und nicht mehr direkt an die Ziel-Web-Server stellt. Da all dies auf der Applikationsebene stattfindet, müssen dementsprechend angepasste Clientprogramme existieren und auch die Sicherheitsanforderungen auf dieser Ebene erfüllt werden.

3.2.4 Kombinierte Techniken

Wie die vorangegangenen Kapitel gezeigt haben, bietet – vor allem reflektiert auf den Anwendungsfall des SBC – keines der genannten Konzepte allein einen umfassenden Lösungsansatz. Im praktischen Einsatz finden sich daher zahlreiche Kombinationsformen (vgl. [Martius, 2000], Seiten 59/60). „In general, the outside filter can be used to protect the gateway from attack, while the inside filter is used to guard against the consequences of a compromised gateway“ (vgl. [Cheswick, 1994], S. 51).

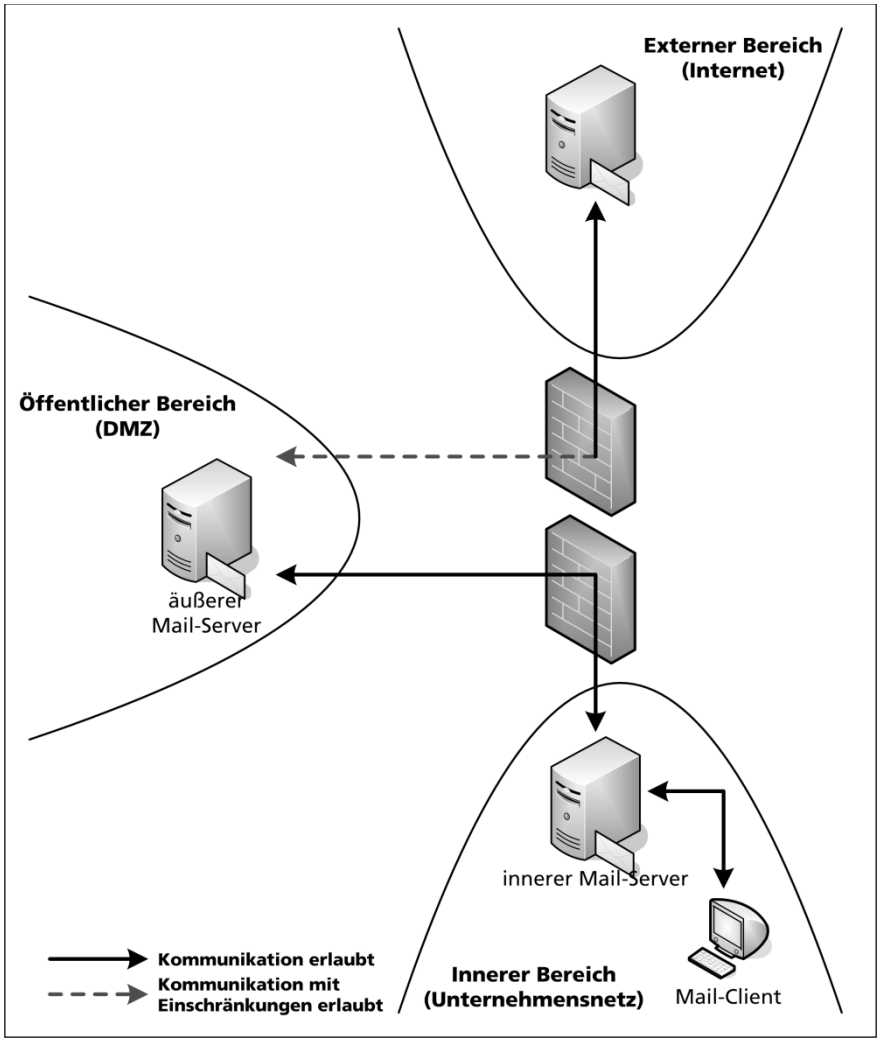
Nach dem Grundprinzip „P-A-P“ (Paketfilter – Application-Level Gateway – Paketfilter) werden die Lösungen entsprechend kombiniert eingesetzt. Im Anwendungsfall des E-Mail-Verkehrs lässt beispielsweise ein externer Paketfilter nur Verbindungen über den TCP-Port 25 (SMTP) zu einem E-Mail Server in der DMZ zu, der als Application-Level Gateway agiert und seinerseits durch den internen Paketfilter ebenfalls nur über TCP-Port 25 mit dem inneren Mail-Server kommunizieren kann (Abbildung 3-5).

Der zunächst naheliegende Schluss, ein ähnliches Konzept einzusetzen, um externen Mitarbeitern den Zugriff auf mittels SBC bereitgestellte Anwendungen zu ermöglichen, darf ohne weitere Überlegungen nicht umgesetzt werden, da ein grundsätzlicher Unterschied in der Art der Anwendungen besteht. Wenngleich natürlich von dem Medium E-Mail in Form von Viren, Trojanern und Spyware eine Gefährdung ausgeht, so wirkt diese asynchrone Kommunikation doch nur mittelbar über den Austausch einzelner Nachrichten, da niemals ein direkter Kontakt zwischen dem externen und dem internen Bereich zustande kommt.

In der Natur von SBC liegt es aber, dass der von extern zugreifende Nutzer eine verbindungsorientierte, interaktive Sitzung mit einem Terminal Server oder virtuellen Desktop öffnen möchte, so dass damit letztendlich die Funktionalität eines Circuit Level Gateways gefordert ist, dem sich – ist die Verbindung erst aufgebaut – die inhaltliche Kontrolle der Sitzung entzieht. Da ein externer Nutzer auf diese Weise Zugriff auf ein System des inneren Bereiches erhält, würde damit das Konzept der voneinander abgegrenzten Sicherheitsbereiche ad absurdum geführt, zumal es nicht um den Zugriff auf eine speziell für einen solchen Zweck ausgelegte Applikation geht sondern um eine interaktive Sitzung mit einem Betriebssystem der Windows-Familie bzw. den dort installierten Anwendungen. Würde dieser Zugriffsweg kompromittiert, so wäre es möglich, den Terminal Server und von diesem aus sämtliche Systeme des inneren Bereiches anzugreifen (vgl. Abbildung 3-6). Es kann nicht als sicher

angesehen werden, dass dies allein durch Maßnahmen der Hostabsicherung lokal auf dem Terminal Server verhindert werden kann.

Abbildung 3-5: Absicherung mittels P-A-P

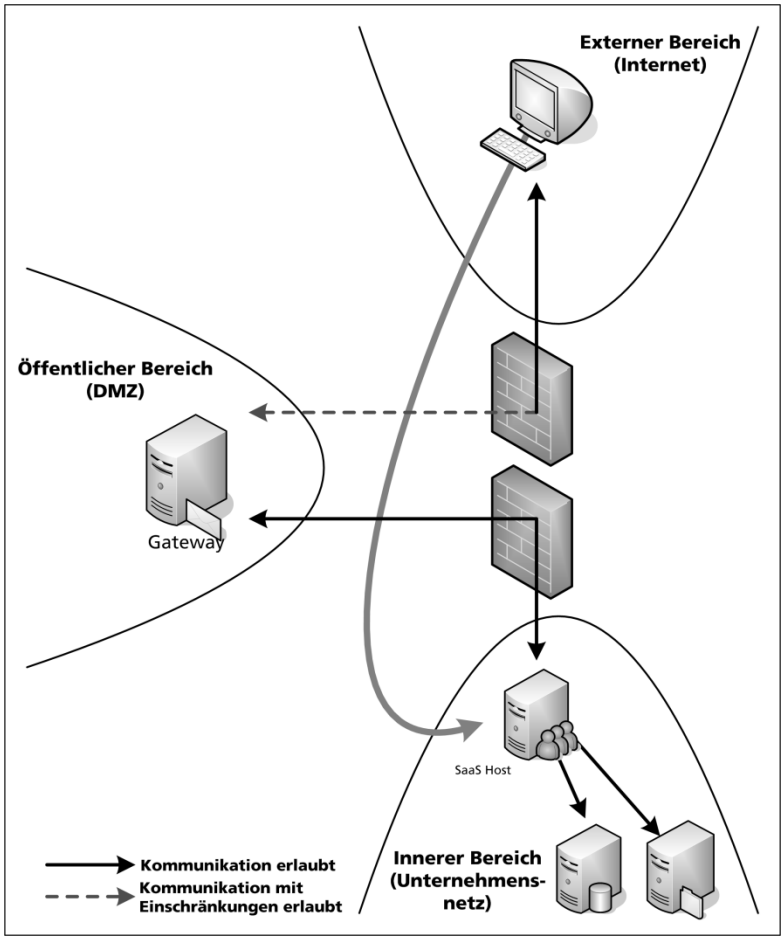


Quelle: Eigene Darstellung

Die vorangegangenen Ausführungen gelten analog für den Einsatz des Terminal Servers in der DMZ mit dem Ziel, Anwendungen an externe Nutzer anzubieten. Der einzige Unterschied besteht hier darin, dass das Gefährdungspotenzial für die übrigen Systeme in diesem Bereich abgestellt werden kann, indem nach dem Prinzip isolierter „security domains“ (vgl. [Cheswick, 1994], S. 53/54) der Terminal Server durch den Einsatz einer zusätzlich Firewall gekapselt wird (vgl. Abbildung 3-7).

Unbeachtet dessen bleibt aber festzuhalten, dass die eingangs definierten Sicherheitsziele für den Terminal Server selbst nicht erreicht werden. Vor allem Authentifizierung und Autorisierung erfordern zusätzliche Maßnahmen, um die Zugriffe aus dem externen Bereich sicher zu gestalten.

Abbildung 3-6: Gefährdungspotenzial für den inneren Bereich



Quelle: Eigene Darstellung

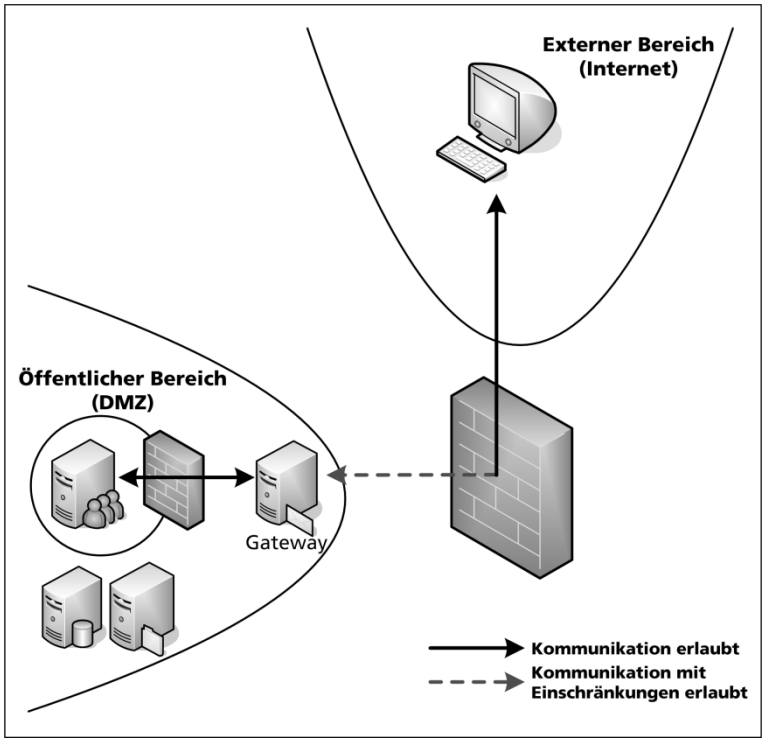
3.2.5 Verschlüsselung

Um den vom externen Bereich ausgehenden potenziellen Gefahren zu begegnen, ist es erforderlich, zwischen dem externen Nutzer und dem Netzbereich, in dem sich der Terminal Server befindet, eine sichere Verbindung – auch als Tunnel bezeichnet – zu etablieren und die komplette Kommunikation gegenüber der unsicheren Umgebung zu verschlüsseln. Unter dem Begriff des Virtual Private Networks (VPN) bieten sich mit Internet Protocol Security (IPSec) oder Secure Socket Layer (SSL) zwei Technologien an, dieses Ziel zu erreichen.

3.2.5.1 Internet Protocol Security (IPSec)

Mittels IPSec werden Sicherheitsfunktionen auf der Netzwerkschicht des ISO/OSI-Referenzmodells realisiert. Dabei besteht die Grundidee darin, jedes einzelne Datenpaket der IP-Schicht zu verschlüsseln und so vor Veränderungen zu schützen. Hierzu kommt eine Kombination von symmetrischen und asymmetrischen Verfahren zum Einsatz.

Abbildung 3-7: Kapselung mittels „Security Domains“



Quelle: Eigene Darstellung

Die eigentliche verschlüsselte Übertragung der Datenpakete erfolgt mittels symmetrischer Verschlüsselung, wofür der Advanced Encryption Standard (AES) mit einer Schlüssellänge von mindestens 256 Bit zum Einsatz kommen sollte (vgl. [Odom, 2008], S. 531ff. und [Schmidt, 2016]).

Dies wirft allerdings die Frage auf, wie die an der Kommunikation beteiligten Partner Kenntnis vom – für diese symmetrische Verschlüsselung notwendigen – gemeinsamen Schlüssel erlangen. Die einfachste Lösung wäre die Verwendung so genannter Pre-Shared Keys (PSK), die von allen Teilnehmern der Kommunikation lokal auf ihren Endgeräten konfiguriert werden müssten. Problematisch ist dabei jedoch insbesondere der Austausch des Schlüssels. Dieser darf unter keinen Umständen über unsichere Wege, wie z. B. unverschlüsselte E-Mails, ausgetauscht werden (vgl. [Odom, 2008], S. 533). Weiterhin sollte der PSK periodisch geändert werden und muss dies sogar umgehend, falls der Verdacht auf eine Kompromittierung bestehen sollte.

Um hier eine Lösung anzubieten, sieht die Architektur von IPSec ein Verfahren namens Internet Key Exchange (IKE) vor, mit dem auf sichere Weise dynamische Schlüssel ausgetauscht werden können. Für den Austausch der Schlüssel finden dabei asymmetrische Verschlüsselungsverfahren Anwendung: „Beim Schlüsselaustausch tauscht man mit seinem Gegenüber auf sicherem Weg einen geheimen (AES-)Schlüssel für die anschließende symmetrische Verschlüsselung der eigentlichen Nutzdaten aus. (…) Das prominenteste und am

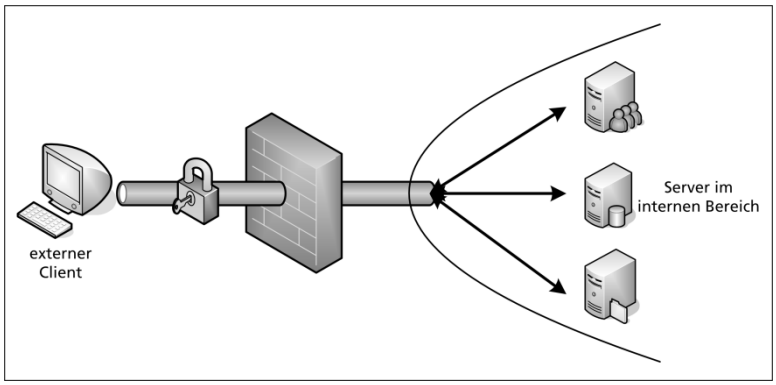
häufigsten genutzte asymmetrische Verfahren ist RSA [benannt nach seinen Entwicklern Rivest, Shamir und Adleman – Anm. d. Verf.], das sowohl für digitale Signaturen als auch für den Schlüsselaustausch zum Einsatz kommt. Bei letzterem hat ihm aber mittlerweile Diffie Hellman (DH) den Rang abgelaufen. Bei DH gehen in jeden einzelnen Schlüsselaustausch Zufallszahlen ein, die die Kommunikationspartner anschließend verwerfen. Somit kann ein Angreifer – anders als bei RSA – den geheimen AES-Schlüssel zu einem späteren Zeitpunkt nicht mehr aus den aufgezeichneten Chiffredaten rekonstruieren (…)" (vgl. [Schmidt, 2016])

Für die asymmetrische Verschlüsselung per DH und RSA sollte nach heutigem Stand der Technik mindestens eine Schlüssellänge von 2.048 Bit zum Einsatz kommen.

Zur Authentifizierung der Client-Computer kann wiederum ein PSK zum Einsatz kommen. Dies erscheint allerdings nur dann sinnvoll, wenn eine begrenzte Menge an Endgeräten zu erwarten ist. Denn auch hier müsste der Schlüssel im Fall einer Kompromittierung auf allen Clients gleichzeitig ausgetauscht werden. Sind eine wachsende Anzahl an Clients und insbesondere auch die Anbindung externer Kunden geplant, bieten sich alternativ Zertifikate an (vgl. Kapitel 3.2.6.3). Dann ist es möglich, diese individuellen Zertifikate pro Client einzeln zu widerrufen und neue auszustellen.

Die verschlüsselte Kommunikation via IPSec erfüllt die Sicherheitsanforderungen der Anonymität, Verbindlichkeit, Vertraulichkeit und Integrität. Ein Vorteil besteht darin, dass dies für die Applikationen der höheren Schichten vollkommen transparent geschieht und somit beliebige Anwendungen geschützt werden können. Mittels eines Tunnels durch die Firewall wird der Client auf IP-Ebene Teil des Zielnetzes (vgl. Abbildung 3-8).

Abbildung 3-8: IPSec-Tunnel



Quelle: Eigene Darstellung

Im Sinne eines abgestuften Sicherheitskonzeptes kann dies aber auch als nachteilig angesehen werden. Ist z. B. das Ziel, über eine solche Verbindung nur Remote-Sitzungen zu erlauben, nicht aber direkten Zugriff auf Datei- oder E-Mail-Server, so muss die IPSec-Verbindung durch Paketfilter auf Port-Basis geeignet eingeschränkt werden. Ein weiterer Nachteil besteht darin, dass beide Endpunkte – auf Seiten des Netzwerks, auf das von außen zugegriffen werden soll, ist dies die Firewall bzw. ein separates Gateway – IPSec implementieren müssen, was für den

externen Nutzer in der Regel bedeutet, dass ein zusätzliches Clientprogramm zu installieren und zu konfigurieren ist.

Dies mag im Falle des Mitarbeiters, der von außen auf das Firmennetzwerk zugreifen möchte, noch zumutbar sein. Bei externen Kunden, die lediglich auf dedizierte SBC Applikationen zugreifen möchten, wird es dagegen kaum auf breite Akzeptanz stoßen, einen Client zur eigentlichen Kommunikation und einen weiteren zur Absicherung derselben zu installieren und zu konfigurieren.

Weiterhin problematisch bleiben die Fragen der Authentifizierung und Autorisierung der Anwender. So muss verhindert werden, dass der extern erreichbare Terminal Server kompromittiert wird bzw. eine nicht autorisierte Person sich eines solchen Hosts bemächtigt. „Der Schutz der (abgespeicherten) Daten auf dem Rechner oder die Zurechenbarkeit von Informationen kann jedoch nur mit Mechanismen auf der Anwendungsebene realisiert werden." (vgl. [Martius, 2000], Seite 96). Über die Sicherheit, die IPSec auf der Netzwerkschicht bietet, hinaus müssen also weitere Schutzfunktionen auf der obersten Ebene des ISO/OSI-Referenzmodells hinzugefügt werden, um alle Sicherheitsanforderungen zu erfüllen (vgl. Kapitel 3.2.6).

3.2.5.2 Secure Socket Layer (SSL)

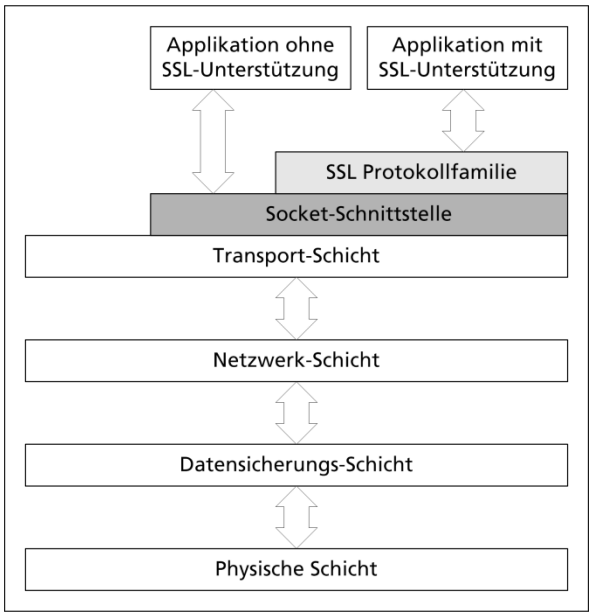
Auch die VPN-Technologie SSL erfüllt die Sicherheitsanforderungen der Anonymität, Verbindlichkeit, Vertraulichkeit und Integrität, wählt dabei jedoch einen anderen Ansatz, nämlich die Implementierung von Sicherheitsfunktionen oberhalb der Transportschicht des ISO/OSI-Referenzmodells. SSL agiert damit als Vermittler zwischen dem TCP/IP-Subsystem und der Anwendungsebene (vgl. Abbildung 3-9).

SSL bedient sich Sockets als Schnittstelle zum TCP/IP-Stack, die von einer Anwendung direkt oder eben durch SSL gesichert angesprochen werden können. Der Vorteil gegenüber IPSec-basierenden Verbindungen besteht darin, dass Sicherheit bis auf die Anwendungsebene etabliert wird. Es kommt (ggf. über ein Gateway als Vermittler) eine verbindungsorientierte Kommunikation von Endpunkt zu Endpunkt zustande. Der Client verbindet sich direkt mit dem Zielserver und nicht mit dem Netzbereich, in dem dieser sich befindet.

Weiterhin entfällt die Notwendigkeit zusätzlicher Clientprogramme – allerdings nur, wenn die abzusichernde Anwendung SSL unterstützt, denn im Gegensatz zu transparenten Verbindungen per IPSec muss die jeweilige Applikation selbst explizite SSL-Aufrufe implementieren. Im Fall reiner Web-Anwendungen ist dies gegeben, denn alle aktuellen Web-Browser unterstützen neben HTTP auch HTTPS – was nichts anderes bedeutet, als das die Kommunikation über den TCP-Port 443 mit SSL bzw. der neueren Transport Layer Security (TLS) verschlüsselt erfolgt. Wenngleich die beiden Techniken SSL und TLS nicht vollends äquivalente Protokolle bezeichnen, findet im Folgenden weiter der Begriff der SSL Verwendung (vgl. [Odom, 2008], S. 538/539).

Analog dazu beinhalten die gängigen Infrastrukturlösungen für den Betrieb von Terminal Servern und virtuellen Desktops Gateways, welche das jeweilige Protokoll zur entfernten Präsentation über den gängigen TCP-Port 443 in SSL getunnelt übertragen. Ebendies leisten beispielsweise der Citrix NetScaler[74], das Microsoft Remotedesktopgateway (RD-Gateway) oder der VMware Horizon Security Server. Auch die Client-Programme der jeweiligen Hersteller können mit der Verschlüsselung der Remote-Protokolle umgehen.

Abbildung 3-9: Einordnung von SSL in das ISO/OSI-Referenzmodell



Quelle: Eigene Darstellung, nach [Martius, 2000], Seite 88

Eine solche Lösung wäre also, bezogen auf mittels SBC bereitgestellte Anwendungen, gleichermaßen für Mitarbeiter wie auch für externe Kunden komfortabel zu handhaben. Client und Server handeln bei der Verbindungsaufnahme ein Verschlüsselungsverfahren aus, das über den kompletten Verlauf der Kommunikation zum Einsatz kommt.

Die Absicherung beliebiger Anwendungen, die nicht für die Kommunikation per SSL vorbereitet sind, ist allerdings nicht ohne Weiteres möglich. Sollen weitere Anwendungen ohne eigene SSL-Implementierung integriert werden, so sind analog zur zuvor beschriebenen Kommunikation per IPSec ein weiteres Programm auf Seiten des Client sowie ein Gateway im Rechenzentrum notwendig, die als Vermittler agieren. Dies leisten beispielsweise der bereits genannte Citrix NetScaler oder die Lösung Cisco AnyConnect. Solche Lösungen kommen aber in der Regel nur zur Versorgung unternehmensinterner Kunden in Betracht, da die zusätzliche Client-Software zu installieren und konfigurieren ist.

[74] Der Citrix NetScaler ist als physische oder virtuelle Appliance verfügbar und löst die frühere Software-Lösung Citrix Secure Gateway (CSG) sowie die Hardware-Appliance Citrix Access Gateway (AG) ab.

Auch bei der Verwendung von SSL erfolgt die Kommunikation mittels symmetrischer Verschlüsselung, wobei typischerweise das Verfahren AES zum Einsatz kommt. Bei der Verbindungsaufnahme authentifiziert sich der Server gegenüber dem Client mit einem Zertifikat und *kann* dies im Gegenzug auch vom Client verlangen. Dies setzt allerdings Zertifikate von externen Anbietern oder eine firmeneigene Public Key Infrastructure (PKI) voraus, womit auch hier weitere Technologien erforderlich werden (vgl. Kapitel 3.2.6.3).

Der Client übermittelt dem Server daraufhin eine Zufallszahl, die mit dem öffentlichen Schlüssel des Servers verschlüsselt wurde, oder die Kommunikationspartner setzen auch hier das asymmetrische Verfahren nach Diffie Hellman (DH) für den Schlüsselaustausch ein.

Es gelten die gleichen Überlegungen wie bei der Kommunikation per IPSec. So ist für AES nach derzeitigem Stand der Technik eine Schlüssellänge von 256 Bit vorzusehen und für DH eine Länge von mindestens 2.048 Bit. Weiterhin ist zu bedenken, dass auch hier durch die Verwendung von Zertifikaten höchstens eine gegenseitige Authentifizierung von Client*computer* und Server möglich wird. Der Gefahr, dass der externe Client kompromittiert wird oder ein nicht autorisierter User sich eines solchen Client bemächtigt, muss gesondert Rechnung getragen werden.

3.2.6 Authentifizierung

Eine Sicherheitsanforderung, die von den bisher vorgestellten Sicherheitslösungen nicht oder nur unzureichend adressiert wird, ist die der Authentifizierung der Endanwender. Die Authentifizierung, also der Beweis der Identität, bildet die Grundvoraussetzung für die Autorisierung, so dass ein Nutzer nur Zugang zu den Informationen erhält, die für ihn bestimmt sind. Autorisierter Zugriff muss natürlich in der Gesamtheit eines Systems berücksichtigt werden, z. B. durch Vergabe entsprechender Zugriffsrechte auf Anwendungsebene. Die Sicherheit beginnt aber bereits beim ersten Kontakt eines Benutzers mit dem System, der Anmeldung. Um die Identität festzustellen, kommen im Regelfall Verfahren nach den Grundprinzipien „something you know“, „something you have“ und „something you are“ zum Einsatz (vgl. [Cheswick, 1994], Seite 119).

3.2.6.1 Passwörter

Als einem jeden Benutzer vertrautes Werkzeug aus dem Bereich „something you know“ kommen sehr verbreitet Benutzeraccount-spezifische Passwörter als Zugangsschutz auf der Anwendungsebene zum Einsatz. Dazu existieren Richtlinien, die eine höhere Sicherheit garantieren sollen, wie die systemseitige Aufforderung, das Passwort periodisch zu ändern, oder Komplexitätsanforderungen, die z. B. einen Mix von Großbuchstaben, Kleinbuchstaben und Zahlen oder Sonderzeichen für ein gültiges Passwort vorschreiben. Da Passwörter erraten sowie auf akustischem oder elektronischem Weg mitgehört werden können, sind sie nur als zusätzliche Komponente in einem Sicherheitskonzept geeignet, kaum aber, um die Identität eines über ein öffentliches Netz zugreifenden Benutzers sicher festzustellen.

3.2.6.2 Token

Als Lösung nach dem Prinzip „something you have" stellen Token als separate Hardware einmalig benutzbare Passwörter, sogenannte „One Time Passwords (OTP)", bereit. Auf Seiten des sicheren Netzes muss ein besonders abzusichernder Server als Gegenstelle existieren, der das vom Token generierte Passwort verifiziert. Dies ist auf zwei Arten möglich. Zum einen können Server und Token jeweils nach dem gleichen (geheimen) Algorithmus über die Zeit variable und nur begrenzt gültige Passwörter generieren, die nach Eingabe durch den Benutzer verglichen werden. Voraussetzung ist in diesem Fall, dass die Zeit zwischen Token und Server synchronisiert ist. Zum anderen kann die Authentifizierung nach dem „Challenge-/ Response-Verfahren" erfolgen.

Der Server stellt dabei einen Code bereit, auf dessen Basis das Token über einen ebenfalls geheimen Algorithmus eine Antwort generiert. Dieses Vorgehen erfordert allerdings aufwändigere Token, da diese über eine eigene Tastatur zur Eingabe des Codes verfügen müssen. Des Weiteren ist das Verfahren für den Benutzer weniger komfortabel, da dieser zwei Codes eingeben muss. Nichtsdestotrotz haben sich Token als Sicherheitslösung im Unternehmensumfeld etabliert. Die Gefahr des Diebstahls eines solchen Tokens kann minimiert werden durch die Kombination mit einer PIN und ggf. einem herkömmlichen Passwort. Es entsteht eine zweistufige Sicherheitslösung mit den Komponenten „something you know" *und* „something you have".

Für das Ziel, Mitarbeiter von extern an ihr Unternehmensnetzwerk anzubinden, ist diese Lösung sehr gut geeignet, da es sich um eine definierte, endliche Menge externer Nutzer handelt. Eine Lösung, die auf zusätzlicher Hardware im Besitz des Nutzers beruht, wird aber für den Zweck, externen Kunden mittels SBC veröffentlichte Anwendungen anzubieten, nicht praktikabel sein. Die Erfahrung zeigt, dass eine Kundenbindung nur erfolgen kann, wenn die Bereitstellung des Zugangs spontan möglich ist. Da bei einem erfolgreichen SBC Angebot mit einer variablen und vor allem schnell wachsenden Zahl externer Nutzer gerechnet wird, wird es kaum möglich sein, alle Interessenten zeitnah mit Token auszustatten, zumal die Übergabe dieser Token selbst wiederum auf sichere Weise geschehen muss. Den Vorteil, Applikationen nach dem SBC Prinzip *sofort* bereitstellen zu können, würde eine auf diese Weise abgesicherte Umgebung damit einbüßen.

3.2.6.3 Zertifikate

Wie bereits im Kapitel zur VPN-Technik auf Basis von SSL angeführt, können Zertifikate sowohl an Computer als auch an einzelne Benutzer vergeben werden. Sie dienen dem Zweck, Authentifizierung, Vertraulichkeit und Integrität über den Austausch asymmetrischer Paare öffentlicher und privater Schlüssel (private key/public key) zu gewährleisten. Dabei kann der öffentliche Schlüssel seiner Funktion nach auch als Schloss verstanden werden. Der Absender einer Nachricht verschlüsselt diese mit dem bekannten, öffentlichen Schlüssel des Empfängers. Nur der Empfänger kann diese Nachricht mit Hilfe seines privaten Schlüssels wieder entschlüsseln, wodurch die Vertraulichkeit der Kommunikation erreicht wird. Umgekehrt kann der Absender seine Nachricht mit seinem privaten Schlüssel signieren. Diese elektronische

Signatur lässt sich anhand des öffentlichen Schlüssels des Absenders verifizieren, so dass die Integrität der Kommunikation gewährleistet wird und eine sichere Authentifizierung des Absenders möglich wird. Die Zertifikate werden von einer übergeordneten Zertifizierungsinstanz – Certification Authority (CA) – ausgestellt, „die bestätigt, dass der öffentliche Schlüssel zum vorgeblichen Inhaber gehört" (vgl. [Martius, 2000], Seite 78). Für ein System zum Management der Zertifikate hat sich der Begriff der Public Key Infrastructure (PKI) etabliert.

Neben der Verwaltung und logischen Zuordnung der Zertifikate stellt sich die Frage, wo diese beim Nutzer physisch gespeichert werden. Smartcards als Speicher für ein solches Zertifikat setzen beim Nutzer das Vorhandensein eines Lesegerätes als zusätzliche Hardwarekomponente voraus, so dass hier die gleichen Überlegungen wie bei den zuvor genannten Token gelten: Die *sofortige* Bereitstellung von Anwendungen an eine große Anzahl externer Benutzer würde verhindert, so dass die Zertifikate höchstens Software-basiert zum Einsatz kommen können. Dies weicht jedoch das Prinzip „something you have" auf, da einem Benutzer die Sicherungspflicht und deren Notwendigkeit für eine Datei schwerer zu verdeutlichen sind, als dies bei einer Hardwarekomponente der Fall ist. Ein auf dem zugreifenden Client gespeichertes Zertifikat kann zudem eher kompromittiert werden, so dass es für sich allein nicht als zweifelsfreier Beweis der Identität angesehen werden darf.

Als problematisch erweist sich zudem die Tatsache, dass die übergeordnete Zertifizierungsinstanz dem Clientprogramm des externen Nutzers bekannt sein muss. Die Root-Zertifikate der Zertifizierungsinstanzen zahlreicher kommerzieller Anbieter sind zwar bereits in gängigen Clientprogrammen integriert. Es bleibt aber im Einzelfall zu prüfen, inwieweit diese bei der Anforderung eines Zertifikates durch einen externen Nutzer tatsächlich dessen Identität feststellen. Oftmals geschieht dies lediglich anhand einer E-Mail-Adresse. Die zweifelsfreie Authentifizierung des externen Nutzers wird also auch hiermit nicht erreicht.

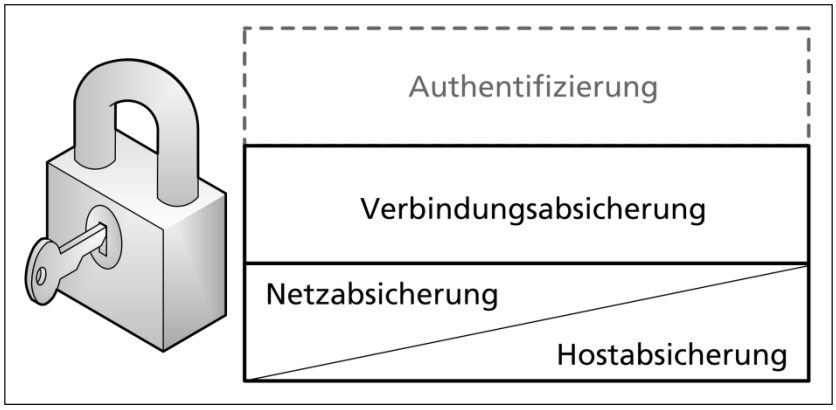
3.2.6.4 Biometrie

Die einzige Methode, tatsächlich die Identität eines externen Nutzers festzustellen, also die Frage nach „something you are", lässt sich nur über biometrische Merkmale feststellen. Eine 100-prozentige Sicherheit lässt sich aber auch hiermit nicht erreichen, da solche Merkmale durch Krankheit oder Alterung Veränderungen unterworfen sind, die das System berücksichtigen müsste. Darüber hinaus setzen sich biometrische Zugangssysteme erst langsam am Markt durch, ohne dass sich bereits ein Standard etabliert hätte. Solche Lösungen sind nach derzeitigem Stand der Technik bislang hochsicherheitskritischen Bereichen, wie z. B. militärischer Anwendung, vorbehalten. Auf einer derart hohen Sicherheitsstufe werden die zu schützenden Informationen typischerweise aber ohnehin in einem separaten Netz gekapselt und selbst durch verschlüsselte Tunnel nicht über öffentliche Netze hinweg ausgetauscht. Allein auf Grund der Tatsache, dass zusätzliche, nach heutigem Kenntnisstand teure und wartungsintensive Zusatz-Hardware erforderlich wäre, werden also nur Lösungen aus den Bereichen „something you know" und „something you have" zum Einsatz kommen können.

Dies gilt insbesondere für das Ziel von SBC Angeboten, eine variable und wachsende Anzahl an externen Kunden mit Applikationen zu versorgen.

Die Diskussion der für SBC in Frage kommenden Sicherheits-Techniken zeigt, dass nicht eine Lösung alleine alle gebotenen Sicherheitsanforderungen von Authentifizierung und Autorisierung über Anonymität bis hin zu Verbindlichkeit, Vertraulichkeit und Integrität erfüllen kann. Es ist vielmehr ein mehrstufiges Sicherheitskonzept erforderlich (vgl. Abbildung 3-10), in dem zunächst über Maßnahmen zur Serverabsicherung (lokale Regelungen auf den Terminal Servern, z. B. durch Berechtigungen im Dateisystem und Systemrichtlinien) und zur Netzabsicherung (Firewalls, Paketfilter) sichergestellt wird, dass Nutzer nur auf die Informationen zugreifen können, zu denen sie autorisiert sind. Über den gezielten Aufbau von Redundanzen (z. B. Server im Lastverbund, Notstromversorgung, redundante Netzwerkanbindung) ist auf dieser Stufe ebenfalls die Konsistenz sicherzustellen. Daraufhin wird dieses Sicherheitskonzept gezielt für einzelne Verbindungen geöffnet, die wiederum im Sinne von Verbindlichkeit, Vertraulichkeit und Integrität abzusichern sind.

Abbildung 3-10: Mehrstufiges Sicherheitskonzept



Quelle: Eigene Darstellung

Als ebenso wichtiger, aber problematischer Baustein erweisen sich Maßnahmen zur sicheren Authentifizierung. Zertifikate einer unternehmensinternen CA sind zur sicheren Identifizierung von Mitarbeitern sehr gut geeignet, da durch persönlichen Kontakt mit einem Vertreter der CA bei der Zertifikatsanforderung die Identität zweifelsfrei festgestellt werden kann. Da dies für externe Kunden eines SBC Angebots nicht ohne weiteres möglich ist und sich zudem die Handhabung der Zertifikate komplizierter gestaltet, kommt diese Technik für SBC Infrastrukturen nur fallweise in Betracht. Zwar gibt es praktische Beispiele, die demonstrieren, dass die Einführung von Zertifikaten möglich ist – z. B. bieten die Sparkassen ihren Kunden unter dem Namen S-Trust[75] die digitale Signatur zur Absicherung an. Jedoch sind in diesem Beispiel eine enge Kundenbindung und ein persönlicher Kontakt gegeben. Dies kann beim Veröffentlichen von Anwendungen über das Internet in der Regel nicht vorausgesetzt werden.

[75] vgl. http://www.s-trust.de (abgerufen am 01.02.2016).

Nach Abwägung der denkbaren zusätzlichen Sicherheitsmaßnahmen verbleibt somit zunächst nur das Passwort, so dass eine Unsicherheit bei der Authentifizierung der externen Nutzer nicht ausgeschlossen werden kann. Auf Grund der genannten Schwächen muss dieses Passwort zumindest durch organisatorische Maßnahmen gestärkt werden, indem dem Benutzer das initiale Passwort nur auf sicherem Wege und seine Sicherungspflichten inklusive Empfehlungen für den Aufbau eines sicheren Passworts explizit bekannt gegeben werden. Zusätzliche Maßnahmen, wie Zertifikate, Smartcards oder Token, die mit Komfortverlust auf Seiten des Nutzers und auch Mehrkosten auf Seiten des Anbieters einhergehen, würden erst dann durchsetzbar und dem externen Nutzer plausibel, wenn mehr als die Bereitstellung von weniger sicherheitskritischen Applikationen geplant ist und mittels SBC benutzerbezogene bzw. höchst vertrauliche Daten zugänglich würden.

3.3 Terminaldienste und Desktop-Virtualisierung

In die im Kapitel 2.6 beschriebene Architektur einer Umgebung zur Anwendungsbereitstellung lassen sich die zuvor genannten Technologien einfügen, indem diese zwischen der Umgebung selbst und den Clients implementiert werden (Abbildung 3-11). Typischerweise wird dazu das Webinterface bzw. der Zugangsserver, der die Clients mit Informationen zu den zur Verfügung stehenden Ressourcen versorgt, hinter dem äußeren Paketfilter in der Demilitarisierten Zone (DMZ) platziert und per Serverzertifikat SSL gesichert, so dass beim Zugriff eine Ende-zu-Ende-Verschlüsselung der übertragenen Anmeldeinformationen gegeben ist.

Der Webserver wird ergänzt um ein Application-Level Gateway, das die Protokolle der entfernten Präsentation, beispielsweise RDP oder ICA, in SSL tunnelt, so dass auch für diese Verbindung über die Verschlüsselung des jeweiligen Remote-Protokolls selbst hinausgehende Sicherheit erreicht wird.

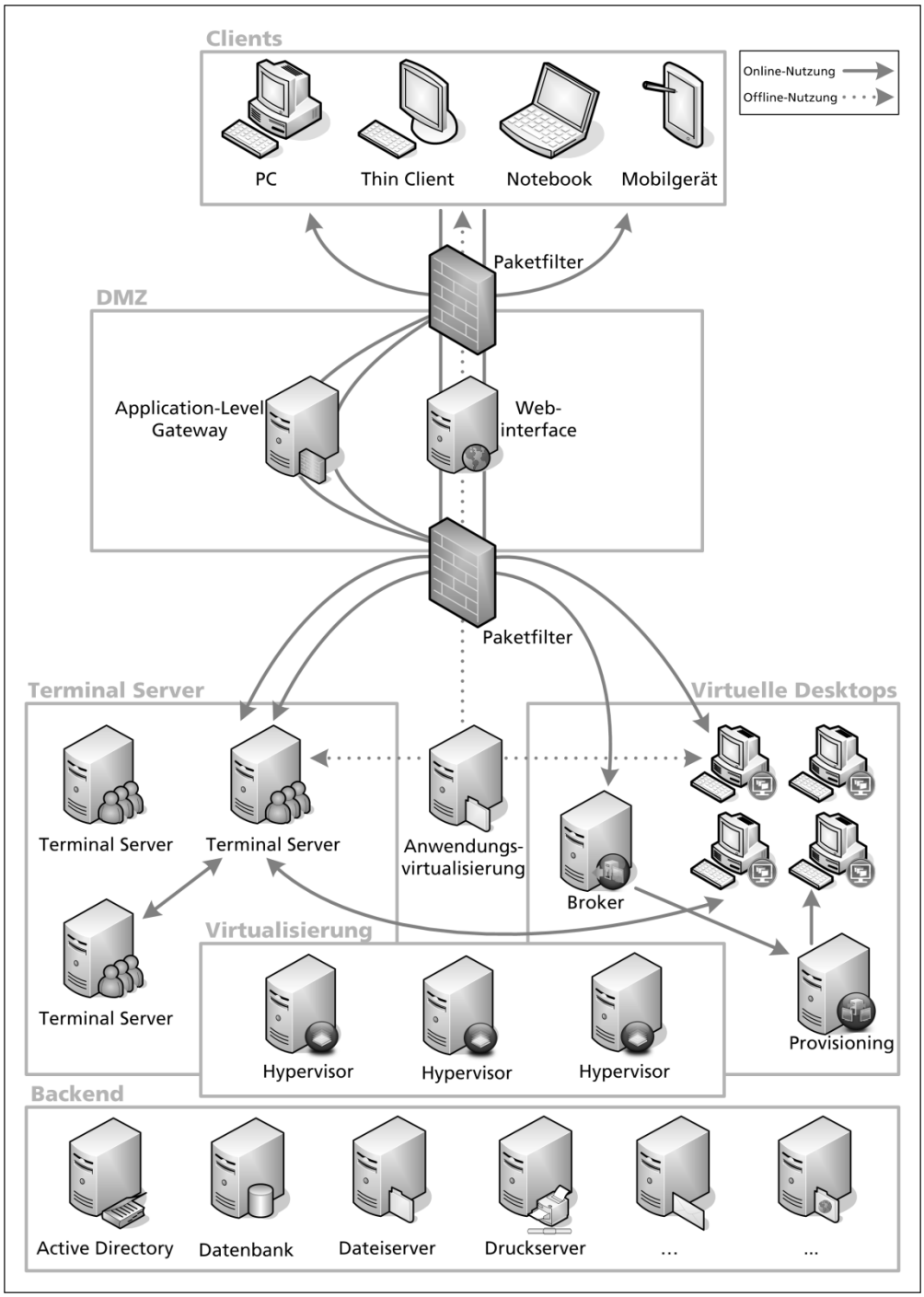
Die SSL-Verbindung terminiert in der DMZ. Von dort greift das Gateway direkt über die Ports des jeweiligen Remote-Protokolls auf die Terminal Server oder virtuellen Desktops zu, die sich hinter dem inneren Paketfilter befinden. Alternativ bzw. zusätzlich sind auch Implementierungen möglich, bei denen auch der Webserver nicht direkt von den Clients kontaktiert wird, sondern sich ebenfalls hinter dem Application-Level Gateway befindet. Darüber hinaus sind heutige Gateway-Lösungen längst nicht auf die Übertragung von Remote-Protokollen beschränkt, sondern können, soweit gewünscht, auch den Zugang zu weiteren Ressourcen, beispielsweise den Zugriff auf Dateifreigaben realisieren.

Unabhängig davon, ob die Infrastruktur in der zuvor erläuterten Weise für den Zugriff von extern ertüchtigt wird oder nur internen Benutzer zur Verfügung steht, sind auf den Terminal Servern und virtuellen Desktops weitere Maßnahmen zur Absicherung der Systeme zu treffen. Das Bundesamt für Sicherheit in der Informationstechnik (BSI) geht in seinen IT-Grundschutz-Katalogen explizit auf die Sicherheit sowohl von Terminal Servern als auch von Virtualisierung im Allgemeinen ein.

Die Virtualisierung von Desktops wird dabei allerdings noch nicht als separater Punkt adressiert. Im Hinblick auf Terminal Server ergänzt das BSI seine Empfehlungen für allgemeine

Server und allgemeine Clients um spezielle Handreichungen für die zentrale Bereitstellung von Ressourcen (vgl. [BSI, 2011], Baustein B 3.305, S. 207 ff.). Die empfohlenen Maßnahmen reichen von der Planung und Konzeption über Beschaffung, Inbetriebnahme und Betrieb bis hin zur Aussonderung eines Terminal Servers.

Abbildung 3-11: Sicherer Zugriff auf eine SBC Infrastruktur



Quelle: Eigene Darstellung, Erweiterung von Abbildung 2-30 um DMZ

Im Kontext dieser Arbeit erscheinen insbesondere die Maßnahmen mit dem Fokus auf „Restriktiver Rechtevergabe auf Terminalservern“ (vgl. [BSI, 2011], Maßnahme M 5.163, S. 3777 ff.) sowie der „Sicheren Nutzung eines Terminalservers aus einem entfernten Netz“ (vgl. [BSI, 2011], Maßnahme M 5.164, S. 3781 ff.) relevant. Letzterer Aspekt wurde bereits im vorangegangenen Kapitel bis hin zur Nutzung von Token zur Erzeugung von OTP ausführlich behandelt, so dass im Folgenden der Schwerpunkt auf der restriktiven Rechtevergabe liegen soll.

Der Maßnahmenkatalog des BSI formuliert allgemeine Punkte, wie z. B. dass auf Terminal Servern grundsätzlich Virenscanner installiert sein sollten, dass ein Terminal Server niemals gleichzeitig auch als Active Directory Domain Controller dienen sollte und dass Anwendungen, die unterschiedliche Anforderungen bezüglich ihres Schutz- und Sicherheitsniveaus mit sich bringen, jeweils auf separaten Terminal Servern betrieben werden sollten. In Bezug auf das Dateisystem fordert der Maßnahmenkatalog unter anderem, dass Zugriffe auf nicht benötigte Dateien unterbunden werden, Software nur von Administratoren installiert werden darf und administrative Programme nur von dazu autorisierten Benutzern ausgeführt werden dürfen. Bei höherem Schutzbedarf soll generell eine Whitelist geführt werden, so dass Endanwender nur vom Administrator freigegebene Applikationen starten können. Leider beschreibt der Maßnahmenkatalog nur, *was* zu tun ist, konkretisiert aber nicht, *wie* die entsprechenden Sicherheitsmaßnahmen umgesetzt werden können. Zum tieferen Verständnis der SBC Betriebsmodelle sollen daher im Folgenden konkrete Möglichkeiten der Umsetzung im Hinblick auf Remotedesktopdienste unter Microsoft Windows Server Betriebssystemen erläutert werden. Die jeweiligen Optionen sind in der Regel ebenso auf virtuelle Desktops anwendbar.

3.3.1 Gruppenrichtlinien

Zur einheitlichen Konfiguration von mehreren Terminal Servern oder Desktop-Instanzen bieten sich in einer auf Microsoft Betriebssystemen basierenden Infrastruktur die Gruppenrichtlinienobjekte, im Englischen „Group Policy Objects (GPO)“, an[76]. Grundlage für deren Anwendung bildet der Windows-eigene Verzeichnisdienst – das Active Directory (AD) – mit der Möglichkeit, sowohl Computer- als auch Benutzer-Objekte in Organisationseinheiten (Organizational Units (OU)) zu strukturieren. Wird eine Menge von Benutzern oder Computern in eine OU verschoben, so lassen sich darauf ein oder mehrere GPO anwenden, die Konfigurations- und Sicherheits-Optionen enthalten können. Analog zur Registry eines Windows-Systems ist ein GPO in einen Computer-spezifischen und einen Benutzer-spezifischen Bereich untergliedert, deren maßgeblicher Zweck es ist, über sogenannte Administrative Vorlagen Registrierungsschlüssel zu erstellen, zu löschen sowie deren Werte zu setzen. Microsoft selbst bietet eine umfangreiche Basis vorgefertigter Administrativer Vorlagen sowohl für die hauseigenen Betriebssysteme als auch weitere Produkte wie die Microsoft Office Suite

[76] Ein guter Einstieg in die Thematik, Grundlagen und Anwendungsbeispiele finden sich unter http://www.gruppenrichtlinien.de (abgerufen am 01.02.2016).

an. Zudem existieren zwei Varianten einer Beschreibungssprache um Administrative Vorlagen anzupassen oder selbst zu erstellen[77].

3.3.1.1 Administrative Vorlagen

Im Hinblick auf die Absicherung von Terminal Servern und virtuellen Desktops finden sich in den Administrativen Vorlagen zahlreiche „Zero Administration"[78] Funktionen, über die die Möglichkeiten der Endanwender, ihre Arbeitsumgebung anzupassen, nahezu beliebig eingeschränkt werden können. Einige dieser Funktionen seien im Folgenden exemplarisch erläutert:

- **System:** Unter den System-Richtlinien empfehlen sich die Optionen „Zugriff auf Eingabeaufforderung verhindern" sowie „Zugriff auf Programme zum Bearbeiten der Registrierung verhindern". Weiterhin definiert die Richtlinie „Angegebene Windows-Anwendungen nicht ausführen" eine Blacklist unzulässiger Programme. Umgekehrt ist es mit der Richtlinie „Nur zugelassene Windows-Anwendungen ausführen" möglich, eine Whitelist erlaubter Programme zu erstellen. Die beiden letztgenannten Richtlinien sollten allerdings nicht gesetzt werden, da es flexiblere Möglichkeiten gibt, die Menge der ausführbaren Programme zu steuern (vgl. Kapitel 3.3.1.2)

- **Systemsteuerung:** Die Funktion „Zugriff auf die Systemsteuerung nicht zulassen" ist selbsterklärend und sperrt die entsprechenden Funktionen für die von der Richtlinie betroffenen Anwender komplett. Granularer steuerbar definiert die Richtlinie „Angegebene Systemsteuerungssymbole ausblenden" eine Blacklist von Systemsteuerungs-Applets, die gesperrt werden. Alternativ dazu ermöglicht die Richtlinie „Nur angegebene Systemsteuerungssymbole anzeigen" im Umkehrschluss, eine Whitelist zulässiger Systemsteuerungs-Applets zu pflegen. Leider ist es in diesen Richtlinien nicht möglich, aus einer Liste der vorhandenen Applets zu wählen. Vielmehr müssen in einer frei definierbaren Liste gezielt die kanonischen Namen der gewünschten Applets[79] eingetragen werden. Für einige Applets, wie „Anzeige", „Drucker", „Programme und Funktionen" oder auch die „Regions- und Sprachoptionen", ermöglichen weitere Richtlinien, granular einzustellen, welche Registerkarten und Optionen innerhalb der jeweiligen Applets zur Verfügung stehen. Mit Fokus auf der IT-Sicherheit sollten die Richtlinien so eingesetzt

[77] Die ältere Syntax wurde bereits mit Windows NT eingeführt. Entsprechende Vorlagen-Dateien enden auf *.adm. Mit dem Windows Server 2008 wurde eine neue auf XML basierende Syntax eingeführt, deren Vorlagen mit *.admx enden. Da die ältere Syntax aber auch unter aktuellen Betriebssystemen bis hin zum Windows Server 2012 R2 noch verwendbar ist und vielfach als einfacher zu erlernen angesehen wird, ist diese auch heute noch verbreitet. Zu den Unterschieden siehe http://technet.microsoft.com/de-de/library/cc753471(WS.10).aspx (abgerufen am 01.02.2016).

[78] Der Begriff „Zero Administration" wird verwendet für sämtliche Funktionen und Maßnahmen, die darauf abzielen, eine Arbeitsumgebung automatisch vorzukonfigurieren, so dass ein Endanwender keine Einstellungen mehr vornehmen *muss* und ohne weitere Konfigurationsarbeit oder Hilfestellung sofort mit einem System oder einer Anwendung arbeiten kann. Damit einhergehend führt „Zero Administration" aber in der Regel auch dazu, dass der Anwender vorgegebene Einstellungen nicht mehr ändern *kann*, was im Hinblick auf die Durchsetzung von Sicherheitsrichtlinien seitens der Administration auch explizit erwünscht ist.

[79] Canonical Names of Control Panel Items: http://msdn.microsoft.com/de-DE/library/ee330741(v=VS.85).aspx (abgerufen am 01.02.2016).

werden, dass Endanwender keinen Zugriff auf globale Systemeinstellungen und Funktionen, die der Installation von Software dienen, haben.

- **Windows-Explorer:** Über die Funktion „Diese angegebenen Datenträger im Fenster "Arbeitsplatz" ausblenden“ kann eine wählbare Menge von Laufwerken im GUI des Windows-Explorers versteckt werden. Die vorgegebene Auswahl an Kombinationen – beispielsweise nur die Laufwerksbuchstaben A:, B: und C: ausblenden – ist begrenzt, kann aber mittels der Syntax für Administrative Vorlagen beliebig angepasst werden[80]. Bei der Verwendung dieser Richtlinie ist allerdings zu berücksichtigen, dass diese nur gegen versehentliche Zugriffe auf die ausgeblendeten Laufwerke schützt. Gibt ein Anwender gezielt einen Laufwerksbuchstaben in einem Eingabefeld des Explorers ein, so ist es ihm dennoch möglich auf das Laufwerk zuzugreifen. Eine Erhöhung der Sicherheit in dem Sinne, dass Zugriffe tatsächlich unterbunden werden, ist nur möglich, wenn zusätzlich die Richtlinie „Zugriffe auf Laufwerke vom Arbeitsplatz nicht zulassen“ angewendet wird. Hierbei ist in der Praxis abzuwägen zwischen dem geforderten Sicherheitsniveau und Funktionalität sowie Bedienkomfort für die Anwender. Wird beispielsweise der Zugriff auf die Systempartition C: gesperrt, so müssen zuvor Ordner des Benutzerprofils, wie z. B. die Anwendungsdaten, Dokumente, Favoriten und weitere Ordner mit benutzerspezifischen Daten auf einen nicht gesperrten Pfad umgeleitet werden (vgl. [Knermann, 2009b]), da es andernfalls im Benutzerkontext nicht mehr möglich ist, auf Dateien zuzugreifen, die für das ordnungsgemäße Funktionieren einer Benutzersitzung erforderlich sind. Wird die Richtlinie aktiviert, ist es nicht mehr möglich, die betroffenen Laufwerke zu durchsuchen, direkte Verknüpfungen auf ausführbare Dateien funktionieren allerdings noch. Dies gilt auch für Verknüpfungen, die ein Anwender selbst anlegt und dabei gezielt einen ihm bekannten oder vermuteten Pfad zu einem ausführbaren Programm eingibt. Soll der Start bestimmter Programme durch einzelne Benutzer oder Benutzergruppen wirksam unterbunden werden, sind entsprechend weitere Maßnahmen zu ergreifen (vgl. Kapitel 3.3.1.2).

3.3.1.2 Benutzerrechte und Dateisystem

Neben der Kontrolle über Registrierungsschlüssel und deren Werte bieten die GPO zudem die Möglichkeit, zentral Vorgaben für die Kontorichtlinien und die lokalen Richtlinien eines Computers zu treffen. So können über den Punkt „Zuweisen von Benutzerrechten“ global Rechte an Benutzergruppen vergeben oder diesen entzogen werden. Beispielsweise kann einer Benutzergruppe hier das Recht zur lokalen Anmeldung an sämtlichen Systemen, auf die die Richtlinie Anwendung findet, verwehrt werden. Weiterhin ist es möglich, zentral Berechtigungen für Äste und Schlüssel der Registrierung sowie für Ordner und Dateien im Dateisystem der Zielcomputer zu setzen. Von dieser Option sollte allerdings nur für Terminal Server oder eine sehr geringe Anzahl virtueller Desktops Gebrauch gemacht werden, da die entsprechenden GPO periodisch ungefähr alle 18 Stunden erneut eingelesen und angewendet werden und dies innerhalb einer VDI eine starke Belastung im Hinblick auf Eingabe-/Ausgabe-

[80] Using GPO to hide specified drives: http://support.microsoft.com/kb/231289 (abgerufen am 01.02.2016).

Operationen bedeuten kann. In einer solchen Umgebung sollten die gewünschten Einstellungen stattdessen statisch im Master-Image der Desktops gesetzt werden[81]. Sofern das Ziel ist, den Zugriff auf Anwendungsprogramme auf bestimmte Benutzer oder Gruppen zu beschränken, empfehlen sich alternativ oder auch ergänzend zu Berechtigungen im Dateisystem die im folgenden Kapitel erläuterten Maßnahmen.

3.3.1.3 Richtlinien für Softwareeinschränkung und AppLocker

Seit Einführung der Betriebssysteme Microsoft Windows XP und Windows Server 2003 existiert mit den Richtlinien für Softwareeinschränkung – im Englischen als Software Restriction Policies (SRP) bezeichnet – ein nützliches Werkzeug, um die Menge der ausführbaren Programme genau zu steuern (vgl. [Tritsch, 2003], S. 296 ff.). Wird eine solche Richtlinie neu erstellt, lautet die Standardeinstellung zunächst „Nicht eingeschränkt", d. h. die Ausführung sämtlicher Programme ist für alle Benutzer erlaubt. Ein nicht zu restriktiver Grundschutz für einen Terminal Server oder virtuellen Desktop ließe sich hier beispielsweise realisieren, indem die Grundeinstellung auf „Nicht erlaubt" geändert und dann mit Hilfe zusätzlicher Regeln gezielt wieder gelockert wird. In praktischen Szenarien des Autors hat es sich hier beispielsweise bewährt, für 64-Bit Zielsysteme die folgenden Pfad-Regeln als zusätzliche Ausnahmen zu definieren:

- C:\Program Files (x86)*
- C:\Program Files*
- C:\Windows\System32*
- C:\Windows\SysWOW64*
- *.lnk

Eine solche Richtlinie sorgt dafür, dass außer den lokal auf dem Zielcomputer vorhandenen Systemprogrammen und Anwendungen keine weiteren ausführbaren Dateien, insbesondere nicht von Benutzern heruntergeladene Objekte von Netzlaufwerken, gestartet werden können[82]. Wichtig zu beachten ist bei der Anwendung dieser Richtlinie, wie auch allgemein bei allen Richtlinien, die Benutzerrechte beschränken, dass in den Sicherheitseinstellungen des jeweiligen GPO für Administratoren das Recht „Gruppenrichtlinie übernehmen" explizit verweigert wird, um nicht versehentlich den Administratoren die Kontrolle über die Zielsysteme zu entziehen.

Zu berücksichtigen ist weiterhin, dass die zuvor angeführten Pfad-Regeln nur dann ein ausreichendes Maß an Sicherheit bieten, wenn innerhalb der zugelassenen Ordner und aller Unterordner die Dateisystemberechtigungen so konfiguriert sind, dass Benutzer nur lesenden Zugriff haben. Andernfalls wäre es Benutzern möglich, ausführbare Dateien in einen erlaubten

[81] http://www.grouppolicy.biz/2011/11/best-practice-group-policy-for-virtual-desktops-vdi/ (abgerufen am 01.02.2016).

[82] Die pauschale Angabe des Wildcard-Parameters „*.lnk" ist erforderlich, wenn Desktop und Startmenü der Benutzer auf Netzlaufwerke umgeleitet werden, da in einem solchen Szenario ansonsten keine Benutzer-spezifischen Verknüpfungen mehr funktionieren würden. Dies würde Bedienbarkeit und Komfort für die Endanwender deutlich einschränken.

Pfad zu kopieren und von dort zu starten. Sofern es aus Gründen der Kompatibilität nicht zu vermeiden ist, Benutzern auch Schreibrechte zu erteilen, bietet sich als Alternative zur Pfad-Regel an, mit Hash-Regeln zu arbeiten. Eine solche Regel bildet einen Hash-Wert über eine ausführbare Datei, um diese eindeutig identifizieren zu können (vgl. [Tritsch, 2003], S. 299).

Der Vorteil ist, dass auf diese Weise explizit nur bestimmte Dateien erlaubt werden und erkannt wird, wenn diese verändert werden. Andererseits ist als Nachteil festzuhalten, dass Hash-Regeln fortlaufend anzupassen sind, sobald ausführbare Dateien sich auf Grund von erwünschten Updates ändern. Soll ein Regelwerk allein auf Basis von Hashes erstellt werden, ist es weiterhin äußerst komplex, eine vollständige Liste sämtlicher Dateien zu ermitteln, welche durch Benutzer ausführbar sein sollen. Entsprechend empfehlen sich Hash-Regeln nur für Umgebungen mit höherem Schutzbedarf, in denen der eingangs vorgestellte Grundschutz mittels Pfad-Regeln als nicht ausreichend angesehen werden muss.

Tabelle 3-1: Unterschiede zwischen SRP und Applocker

Feature	**Software Restriction Policies**	**AppLocker**
Regelbereich	Alle Benutzer	Bestimmte Benutzer oder Gruppen
Bereitgestellte Regelbedingungen	Dateihash, Pfad, Zertifikat, Registrierungspfad und Internetzonenregeln	Dateihash, Pfad und Herausgeberregeln
Bereitgestellte Regeltypen	Zulassen oder Verweigern	Zulassen oder Verweigern
Standardregelaktion	Zulassen oder Verweigern	Verweigern
Überwachungsmodus	Nein	Ja
Assistent zum Erstellen von mehreren Regeln gleichzeitig	Nein	Ja
Richtlinienimport/-export	Nein	Ja
Regelsammlung	Nein	Ja
PowerShell-Unterstützung	Nein	Ja
Benutzerdefinierte Fehlermeldungen	Nein	Ja

Quelle: nach Microsoft, 2012[83]

Sollen unterschiedliche Gruppen von Anwendern gemeinsam auf einem Terminal Server arbeiten, jedoch jeweils nur für sie freigegebene Applikationen nutzen dürfen, so ergibt sich ein weiterer Nachteil. Innerhalb einer SRP ist es nicht möglich, einzelne Benutzer oder Gruppen zu differenzieren. Ein komplexeres Regelwerk lässt sich entsprechend nicht mit nur einem GPO abbilden. Als Ausweg bleibt, pro Gruppe ein separates GPO zu erstellen und dieses über die Delegierung auf eben diese Gruppe zu beschränken. Eine weitere Option ist eine neue Funktion namens AppLocker, die alle Editionen des Windows Server 2008 R2 und neuer als Alternative zu den SRP beinhalten. AppLocker findet sich in den Einstellungen eines GPO unter dem Begriff der Anwendungssteuerungsrichtlinien wieder. Im Gegensatz zu den SRP erlaubt es

[83] Übersicht zu Windows AppLocker: http://technet.microsoft.com/de-de/library/dd759113.aspx (abgerufen am 01.02.2016).

AppLocker, Regeln auf bestimmte Benutzer oder Gruppen zu beschränken. Weitere Verbesserungen beinhalten unter anderem die Möglichkeiten, Einstellungen zu importieren und zu exportieren sowie benutzerdefinierte Texte für Fehlermeldungen bei Regelverstoß anzugeben (vgl. Tabelle 3-1).

Auf Client-Seite können AppLocker-Richtlinien allerdings neben allen Editionen des Windows Server 2008 R2 nur auf Windows 7 in den Varianten Ultimate und Enterprise angewendet werden[84]. Auch Windows Server 2012 (R2) und die Enterprise-Variante von Windows 8.x unterstützen diese Funktion. Sofern ältere bzw. davon abweichende Betriebssysteme zu verwalten sind, verbleiben als kleinster gemeinsamer Nenner nur die SRP, die auch unter Windows Server 2008 R2 und Windows 7 noch anwendbar sind oder als Alternative zusätzliche Produkte von Drittanbietern. So findet sich in [Königsmann, 2011] neben Handreichungen zur Inbetriebnahme von AppLocker ein exemplarischer Vergleich dessen Möglichkeiten mit einem Zusatzprodukt mit dem Fazit „In speziellen Windows-Umgebungen und für bestimmte Rahmenbedingungen bietet AppLocker einen guten Schutz. Wer aber heterogene und insbesondere ältere Windows-Systeme betreibt, ist mit Software von Drittanbietern besser bedient."

Als Nachteile des AppLocker werden angeführt, dass dieser eine homogene Umgebung bestehend aus den neuesten Microsoft-Betriebssystemen sowie eine Active Directory Domäne zur automatischen Verteilung der Einstellungen voraussetzt. Diese Voraussetzungen dürfen aber im vorliegenden Gegenstand der Untersuchung – einer von den Endanwendern und ihren Client-Computern gekapselten Umgebung zur Bereitstellung von Applikationen mittels SBC – als gegeben angenommen werden, so dass AppLocker zur Absicherung der Terminal Server und virtuellen Desktops als geeignetes Werkzeug angesehen werden kann.

3.3.2 Exkurs: ReCoBS als Sonderfall für Terminaldienste

Das BSI verwendet den Begriff eines „Remote-Controlled Browsers System (ReCoBS)" zur Beschreibung eines Sonderfalls für den Einsatz von Terminaldiensten, bei dem besonders abgesicherte Terminal Server explizit nicht gemeinsam mit den Client-Computern der Endanwender im internen LAN eines Unternehmens betrieben werden. Die Terminal Server werden stattdessen in der DMZ oder einem anderen separaten Netzbereich eingesetzt, der ein niedrigeres Sicherheitsniveau als das interne Netz hat. Die in diesem Netz eingesetzten Terminal Server bieten den Clients im internen Netz auf sichere Weise einen Web Browser als veröffentlichte Anwendung an. Das Ziel dabei ist, sämtliche aktiven Inhalte, wie z. B. JavaScript, Java-Applets, ActiveX-Komponenten oder auch Flash- und Shockwave-Objekte außerhalb des schützenswerten inneren Netzes auszuführen. Auf diese Weise wird gezielt ein Bruch im Informationsfluss herbeigeführt, indem Datenverarbeitung und Benutzerinteraktion getrennt werden. In den IT-Grundschutzkatalogen benutzt das BSI den Begriff eines

[84] Varianten und Versionen von Windows, die AppLocker unterstützen: https://technet.microsoft.com/en-us/library/ee424382.aspx (abgerufen am 01.02.2016).

„Terminalservers als grafische Firewall“ (vgl. [BSI, 2011], Maßnahme M 4.365, S. 3401 ff.), relativiert diese Begrifflichkeit aber in einem separaten Dokument zu den Grundlagen und Anforderungen eines ReCoBS (vgl. [BSI, 2006], S. 8). Demnach soll der Begriff nicht angewendet werden, da es sich bei einem ReCoBS nicht um eine eigenständige Firewall, sondern nur um eine optionale Ergänzung eines mehrstufigen Sicherheitskonzepts handelt, das weiterhin separate Firewalls (vgl. Kapitel 3.2) erfordert, welche das ReCoBS gegen das Internet und das innere Netz gegen das ReCoBS abschotten.

Das ReCoBS dient dabei dem Schutz von internen Netzen mit erhöhtem Sicherheitsbedarf, wobei für behördliche Anwendungen in diesem internen Netz höchstens Daten der Geheimhaltungsstufe „Verschlusssache – Nur für den Dienstgebrauch (VS-NfD)“ verarbeitet werden dürfen. Weiterhin ist zu berücksichtigen, dass die zwischen Internet und ReCoBS übertragenen Daten nicht gesondert geschützt werden und somit das primäre Ziel eines ReCoBS die uneingeschränkte Internet-Recherche ist. Die Übertragung schützenswerter Daten an Web-Anwendungen darf auf diesem Weg entsprechend nicht erfolgen. „Auf dem ReCoBS-Server (…) dürfen keinerlei sensitive Anwendungen betrieben oder sensitive Daten gespeichert werden. Beispielsweise sollte ReCoBS keinen Zugriff auf das **Intra**net erlauben“ (vgl. [BSI, 2006], S. 14).

Vor diesem Hintergrund wäre es im Hinblick auf die Realisierung von SBC Infrastrukturen denkbar, dass eine Unternehmung ein ReCoBS nicht selbst betreibt, sondern dies einem externen Provider überlasst, sofern dieser gewährleistet, dass das ReCoBS den Anforderungen des BSI genügt. Die technischen und organisatorischen Maßnahmen, die dafür Sorge tragen, dass keine sensitiven Daten das interne Netz verlassen, liegen in jedem Fall beim Betreiber des internen Netzes.

In technischer Hinsicht sei angemerkt, dass sich gemessen an den vom BSI formulierten Anforderungen an ein ReCoBS sowohl Terminal Server als auch virtuelle Desktops für die Umsetzung eignen. Insbesondere Techniken, wie z. B. die in Kapitel 2.3 beschriebenen Citrix Provisioning Services, können dabei eine sinnvolle Ergänzung sein, da eine Kompromittierung bzw. das Herunterladen potenziell schädlicher Inhalte auf das ReCoBS als „Opferrechner“ (vgl. [BSI, 2006], S. 14 und S. 19) kurzfristig in Kauf genommen wird. Beim Betrieb von provisionierten Standard-Images lassen sich solche Inhalte mit einem Neustart des ReCoBS-Servers oder -Desktops auf einfache Weise beseitigen.

3.4 (Server-)Virtualisierung

Alle bisher in diesem Kapitel diskutierten Aspekte der IT-Sicherheit sind auf Terminaldienste und Desktops anwendbar unabhängig davon, ob diese auf physischen oder virtuellen Maschinen betrieben werden. Soll aber, wie es beim Anwendungsfall von SBC die Regel sein dürfte, eine Virtualisierungsinfrastruktur als Basis dienen, sind zusätzliche Aspekte speziell in Bezug auf die (Server-)Virtualisierung mittels Typ-1 Hypervisoren zu berücksichtigen. Das BSI empfiehlt hierzu Maßnahmen sowohl technischer als auch organisatorischer Art (vgl. [BSI, 2011], Baustein B 3.304, S. 203 ff.). Diese betreffen zum einen beispielsweise die Frage der ggf. erforderlichen „Aufteilung der Administrationstätigkeiten bei Virtualisierungsservern“ in

größeren Unternehmen (Maßnahme M 2.446), zum anderen konkrete Aspekte der Implementierung, wie die „Einsatzplanung für virtuelle IT-Systeme“ (M 2.444) und deren sichere Konfiguration (M 4.346).

Ein weiteres Augenmerk liegt insbesondere auf der Netzwerkinfrastruktur mit Empfehlungen und Prüffragen zur „Planung des Netzes für virtuelle Infrastrukturen“ (M 5.153) und der „sicheren Konfiguration eines Netzes für virtuelle Infrastrukturen“ (M 5.154). Die Maßnahmen beinhalten unter anderem die Maßgabe, das Management-Netz und auch das Netzsegment für die Anbindung von Storage separiert zu betreiben, um diese besonders schützenswerten Bereiche dem Zugriff aus dem Netzsegment – ggf. kompromittierter – virtueller Maschinen zu entziehen. Gleiches gilt für Virtualisierungsfunktionen, wie z. B. die Live Migration, „da bei einer Live Migration die Hauptspeicherinhalte eines virtuellen IT-Systems möglicherweise unverschlüsselt über das Netz übertragen werden“ (vgl. [BSI, 2011], Maßnahme M 5.153, S. 3748 ff.).

Wenngleich die IT-Grundschutz-Kataloge einen umfassenden Überblick über die notwendigen Sicherheitsmaßnahmen geben, sind die Ausführungen doch sehr allgemein gehalten. Da aber die Virtualisierung insbesondere im Hinblick auf SBC Infrastrukturen einen integralen Bestandteil des Betriebskonzeptes bildet, sollen mögliche Gefährdungen und Gegenmaßnahmen im Folgenden noch detaillierter erläutert werden.

Neben den Angriffsvektoren, denen eine Umgebung mit einem mehrstufigen Sicherheitsmodell, welche aus dem Internet erreichbar ist, ohnehin ausgesetzt ist, eröffnen sich weitere Gefahrenquellen, da die Komplexität des Gesamtsystems ansteigt. „Ein virtualisiert betriebenes System ist daher mindestens genauso angreifbar wie ein nicht virtualisiertes System“ (vgl. [Lenz, 2010], S. 30). Gefahren ergeben sich bereits unter organisatorischen Gesichtspunkten aus der oftmals inflationär wachsenden Anzahl von virtuellen Maschinen. Da der Einsatz einer Virtualisierungsinfrastruktur das Erstellen zusätzlicher Maschinen signifikant vereinfacht, ist es in der Praxis oftmals zu beobachten, dass zu Testzwecken neben den produktiven Systemen zahlreiche weitere VM erstellt werden, die aber ebenso zu warten und mit Updates zu versorgen sind. Der Administrationsaufwand nimmt somit signifikant zu. Gefährdungen durch nicht aktuelle Systeme kann organisatorisch begegnet werden, indem jedwede VM einem koordinierten Lebenszyklus unterworfen wird und entweder regelmäßig mit Updates versorgt oder aber außer Betrieb genommen wird.

Aber auch in technischer Hinsicht sind Maßnahmen zu treffen, denn die Angriffsfläche einer Infrastruktur vergrößert sich durch den Einsatz von Virtualisierung und eine wachsende Zahl von VM, da jede VM grundsätzlich einen potenziellen Schwachpunkt für einen Angriff auf andere VM oder den Hypervisor selbst darstellt. Ist letzterer erst kompromittiert, ist die Infektion des Gesamtsystems, selbst aus nicht kompromittierten VM heraus, kaum mehr feststellbar. So ist eine Manipulation des Hpyervisors durch ein Rootkit ein denkbares Szenario. Eine Klassifizierung von Rootkit-Varianten findet sich in [Lenz, 2010], Kapitel 2.3, ebenso wie Überlegungen zu möglichen Angriffsvektoren. Ziel eines Angriffs könnte es sein, den vorhandenen Hypervisor zu kompromittieren und entweder zu modifizieren oder zu ersetzen

oder aber den vorhandenen „legalen Hypervisor in einen schadhaften einzubetten bzw. zu verschachteln“ (vgl. [Lenz, 2010], S. 26). Dabei handelt es sich nicht nur um ein theoretisch denkbares Szenario. Auch praktisch ist die Verschachtelung mehrerer Hypervisoren tatsächlich umsetzbar (vgl. [Söldner, 2012]) – wenn auch im beschriebenen Fall nicht als Teil eines Angriffs, sondern als explizit gewünschter Aufbau einer Testumgebung. Realistischer ist der Angriff auf eine Gast-VM mit dem Ziel, auf diesem Wege mittelbar Zugriff die Management-VM bzw. den Hypervisor selbst zu erlangen und diesen kompromittieren.

Ohne Techniken wie Intel VT-d und AMD IOMMU (siehe Kapitel 2.2.7) wäre eine solche Kompromittierung des Hypervisors über die Manipulation der Treiber DMA-fähiger Hardware ein durchaus realistisches Angriffsszenario (vgl. [Lenz, 2010], S. 35 ff.), in dessen Verlauf es möglich wäre, die Gerätetreiber von Eingabe-/Ausgabe-Geräten zur Laufzeit zu modifizieren und so den Hypervisor zu kompromittieren. VT-d und IOMMU verhindern das willkürliche Überschreiben geschützter Speicherbereiche, indem jeder DMA-fähigen Komponente ein separater Bereich des Hauptspeichers zugeordnet wird und die Geräte jeweils nur auf diesen Bereich zugreifen können (vgl. [Lenz, 2010], S. 56). Die in Kapitel 6 exemplarisch betrachtete Virtualisierungsumgebung verwendet entsprechend Intel TV-x und VT-d fähige Hardware sowie mit dem Citrix XenServer Enterprise einen Hypervisor, welcher diese Funktionen zwingend voraussetzt.

3.5 Zusammenfassung

Für eine SBC Implementierung, welche auf einer Virtualisierungsinfrastruktur basiert, sind zunächst sämtliche Sicherheitsmaßnahmen zu treffen, die auch auf eine herkömmliche Umgebung mit einem mehrstufigen Sicherheitsmodell anzuwenden sind. Neben der Verwendung einer DMZ und der Absicherung des Netzes nach dem P-A-P Prinzip (vgl. Kapitel 3.2.4) gehören etablierte Update- sowie Patchmanagement-Prozesse und Lösungen zur Abwehr von Viren und Malware unverzichtbar zum Sicherheitskonzept. „Des Weiteren ist bei der Auswahl virtueller Maschinen darauf zu achten, dass Anwendungen und Services nach streng definierten Sicherheitsmaßstäben voneinander getrennt werden. (…) der Betrieb von Produktiv-Servern neben denen einer Testumgebung (…) sollte von vorn herein vermieden werden. Auch wenn Plattform-Virtualisierung die Konsolidierung von Hardware unterstützt, so ist die Ressourcen-Einsparung nicht selbst erklärtes, sondern ein der Sicherheit unterzuordnendes Ziel“ (vgl. [Lenz, 2010], S. 40).

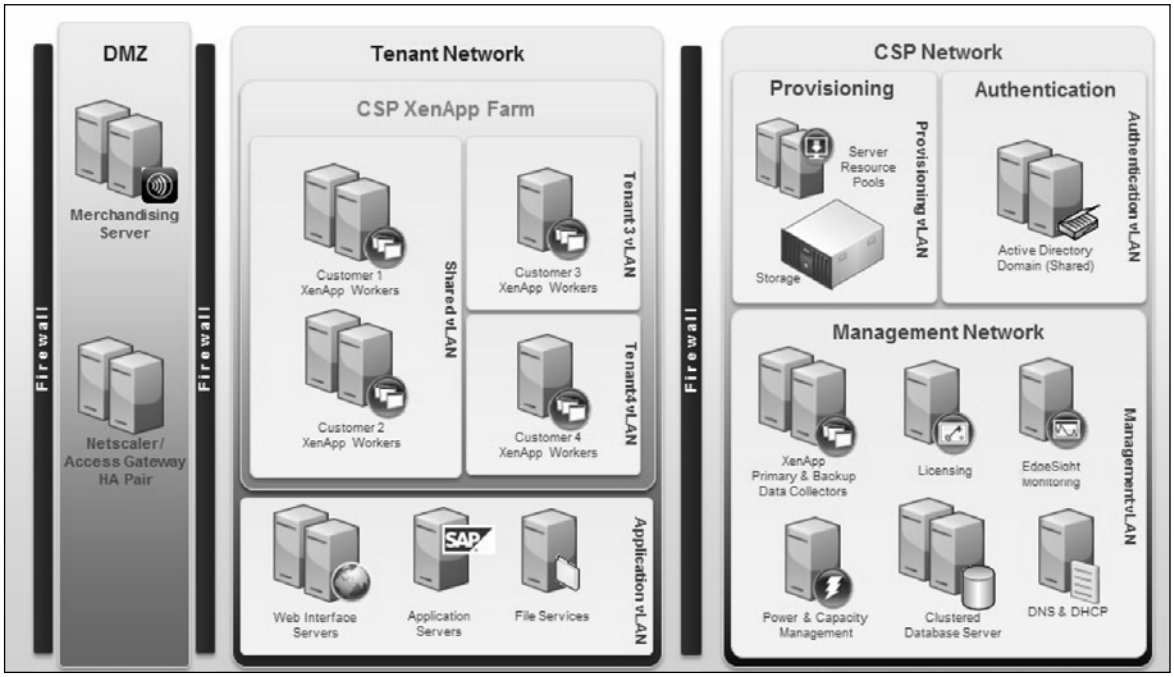
Entsprechend sollten die verschiedenen Sicherheitszonen – d. h. die DMZ sowie das interne Netz – physisch getrennte Hardware und somit separate Pools von Hypervisoren verwenden[85]. Sofern im Rahmen von SBC Kunden mit unterschiedlichen Sicherheitsbedürfnissen zu bedienen sind, sollten auch die jeweiligen Systeme in getrennte Umgebungen aufgeteilt werden. Ein weiterer Aspekt betrifft den Hypervisor selbst, der ebenso wie die Betriebssysteme der VM und

[85] „In Public-Cloud-Umgebungen ist diese Trennung nicht möglich. Konkret werden hierbei Systeme verschiedener Kunden innerhalb einer dynamisch aufgeteilten Infrastruktur betrieben. Bei der Planung von Cloud-Umgebungen gilt es daher, einen Kompromiss zwischen Flexibilität, Auslastung und Sicherheit zu finden.“ (vgl. [Lenz, 2010a], S. 20).

die bereitzustellende Anwendungssoftware, in das Updatemanagement einzubinden ist. Ferner sollte ein separates physisches Netzwerk-Interface exklusiv für das Management des Hypervisors genutzt werden. Dieses darf nicht von den VM aus zugreifbar sein (vgl. [Lenz, 2010a], S. 72).

In der Praxis sind allerdings auch Infrastrukturen anzutreffen, die auf eine vollständige Trennung von Kunden verzichten. So hat Citrix eine Referenz-Architektur für eine mandantenfähige Terminal Server Implementierung vorgestellt (vgl. [Citrix, 2011]), die explizit auch die Option beinhaltet, dass mehrere Kunden gemeinsam auf ein und demselben Terminal Server arbeiten. Dieses als „Shared model" bezeichnete Betriebsmodell kommt demnach vor allem bei kleineren Providern zum Einsatz, für die Kosten und nicht Sicherheitsaspekte die wichtigste Rolle spielen (vgl. [Citrix, 2011], S. 13). Auf Grund der zuvor ausgeführten Sicherheitsaspekte wird dieses Modell im Folgenden jedoch nicht weiter betrachtet. Auch unter der Prämisse, dass Kunden mit identischen Sicherheitsbedürfnissen zu bedienen sind, wird mindestens ein „Partial isolation model" vorausgesetzt, bei dem unterschiedliche Kunden zwar gemeinsame Infrastrukturdienste verwenden, d. h. eine gemeinsame Active Directory Gesamtstruktur und eine gemeinsame Terminal Server bzw. Desktop Farm nutzen, jedoch pro Kunde dedizierte Terminal Server bereitgestellt werden, die je nach Schutzbedarf auch in getrennten Netzwerksegmenten bzw. auf separaten Virtualisierungshosts betrieben werden können (vgl. Abbildung 3-12).

Abbildung 3-12: Mandantenfähige Referenzarchitektur



Quelle: [Citrix, 2011], S. 9

Unterschiedliche (Sicherheits-)Einstellungen pro Kunde lassen sich in einer solchen Infrastruktur verwalten, indem pro Kunde eine separate Organisationseinheit im Active Directory oder gar

eine eigene Domäne innerhalb der Active Directory Gesamtstruktur angelegt wird, so dass Gruppenrichtlinien und weitere Berechtigungen pro Kunde gepflegt werden können (vgl. [Citrix, 2011], S. 27). Das Netzwerk des Providers mit den Management-Funktionen wird dabei jedoch in jedem Fall als der Bereich mit dem höchsten Schutzbedarf hinter einer separaten Firewall betrieben.

4 Kapazitätsmanagement: Allgemeine Aspekte und Ziele

Die vorangegangenen Kapitel haben umfassend die aktuell verfügbaren Technologien zur zentralen Bereitstellung von Applikationen und Desktops sowie deren verschiedene Nutzungsmodelle erläutert. In den folgenden Kapiteln sollen nun die Kapazität von SBC Infrastrukturen sowie insbesondere Modelle und Methoden zum Kapazitätsmanagement beim Aufbau neuer SBC Infrastrukturen im Vordergrund stehen. Allgemein gefasst beschäftigt sich das Kapazitätsmanagement mit der Frage, mit wie vielen und welchen Ressourcen – z. B. Anzahl an Virtualisierungshosts sowie Prozessoren pro Host, Menge an Hauptspeicher, Typ und Kapazität des Storage – eine Infrastruktur ausgestattet werden muss, um eine bestimmte Last erfolgreich bewältigen zu können. Aus dieser Fragestellung ergeben sich sogleich weitere Fragen, nämlich zum einen, was eine „bestimmte Last" überhaupt ist und wie sich diese ermitteln lässt, und zum anderen, welche Anforderungen zu erfüllen sind, damit die Last als „erfolgreich bewältigt" betrachtet werden kann (vgl. [Knermann, 2008c], S. 83).

Diese Fragen gilt es zu beantworten unabhängig davon, ob die Anwendungen und Desktops an unternehmensinterne Anwender oder externe Kunden bereitgestellt werden sollen. Wie bei jedem IT-System, mit dem Endanwender unmittelbar interagieren, muss als oberste Prämisse gelten, dass die Benutzer der Infrastruktur flüssig arbeiten können. Es muss also eine direkte Reaktion von Anwendungen und Desktops auf Mausbewegungen und Tastatureingaben erfolgen, wie auch das verzögerungs- und ruckelfreie Scrollen von Bildschirminhalten gegeben sein muss. Mögen diese Anforderungen ausgehend von der lokalen Ausführung von Programmen an einem Desktop-PC als selbstverständlich erscheinen, kommt ihnen insbesondere bei der Bereitstellung von Anwendungen und Desktops im Rahmen der entfernten Präsentation erhöhte Bedeutung zu. Wie die Erfahrungen aus zahlreichen Terminal Server Projekten des Autors zeigen, wirkt sich schlechtes Antwortzeitverhalten einer Benutzersitzung unmittelbar negativ auf die Akzeptanz seitens der Benutzer aus. „Macht der Benutzer hier für einen auch nur verhältnismäßig kurzen Zeitraum schlechte Erfahrungen, so zeigen es Praxisbeispiele leider immer wieder, wird er die Verwendung von Thin Client Technologie und Server Based Computing ablehnen und Möglichkeiten suchen, mit einem lokalen Arbeitsplatz PC arbeiten zu können. Ein Benutzer ist erstaunlicherweise hier kritischer gegenüber dem Server Based Computing, als dem eigenen, ggf. genauso überforderten Arbeitsplatz PC. Es gibt jedoch auch positive Beispiele (…), bei denen die Anwender aufgrund der überlegenen Performance des Terminalservers dort freiwillig auf ihren PC verzichtet haben und lieber mit einem Thin Client arbeiten" (vgl. [Knermann, 2008c], S. 84).

4.1 Quality of Service (QoS)

Um die zuvor umschriebenen Anforderungen konkretisieren und quantifizieren zu können, können folgende Kriterien herangezogen werden, die unter dem Oberbegriff der „Quality Of Service (QoS)" zusammengefasst werden (vgl. [Menascé, 2004], S. 12 ff.):

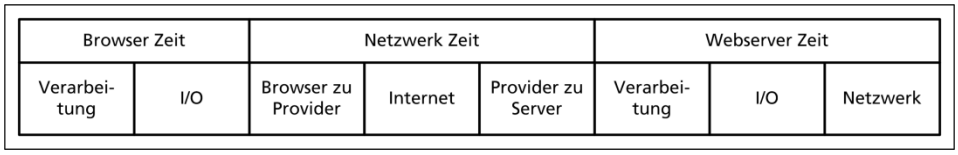
- Antwortzeit
- Durchsatz
- Verfügbarkeit
- Zuverlässigkeit
- Sicherheit
- Skalierbarkeit
- Erweiterbarkeit

Diese werden in den folgenden Absätzen im Hinblick auf SBC Infrastrukturen erläutert.

4.1.1 Antwortzeit

Die Antwortzeit beschreibt die Zeitspanne, die ein IT-System benötigt, um auf eine Anfrage seitens eines Endanwenders zu reagieren. In [Menascé, 2004], Kap. 1.2.1 wird dies am Beispiel eines einfachen Webservers – in diesem Fall einer E-Commerce Seite – erläutert und dargestellt, dass sich bei der Interaktion mit einem IT-System über Weitverkehrsnetze die Antwortzeit aus mehreren Komponenten zusammensetzt, nämlich der Client-lokalen Verarbeitung, der Übertragung über private und öffentliche Netze hinweg sowie der Server-seitigen Verarbeitung (vgl. Abbildung 4-1).

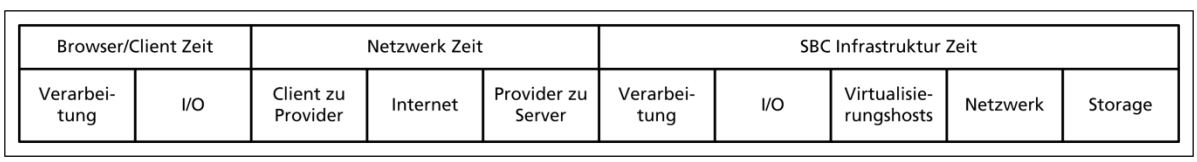
Abbildung 4-1: Antwortzeit, Webserver



Quelle: Eigene Darstellung, nach [Menascé, 2004], S. 13

Dieses Beispiel lässt sich direkt auch auf die Kommunikation mit einer SBC Infrastruktur übertragen, denn auch in diesem Fall beginnt eine Benutzersitzung i. d. R. mit der Kommunikation zwischen Client und einem Webserver – sei es ein Browser, der die für einen Benutzer bereitgestellten Anwendungen in Form einer Webseite aufbereitet, oder ein SBC Clientprogramm, das die verfügbaren Ressourcen von einem Webservice abruft. Analog verläuft beim Verbindungsaufbau zu einer Anwendung oder einem Desktop die Kommunikation zwischen SBC Client und dem Gateway, welches über ein Remote Protokoll die sichere Verbindung in das interne Netz der SBC Infrastruktur herstellt.

Abbildung 4-2: Antwortzeit, SBC Infrastruktur



Quelle: Eigene Darstellung, nach [Menascé, 2004], S. 13

Ist diese Verbindung sodann etabliert, ist weiterhin die Antwortzeit bei der Interaktion mit Anwendungen im Rahmen der entfernten Präsentation zu berücksichtigen. In diesem Fall ist das Modell zu erweitern, um der Komplexität der SBC Infrastruktur im Hinblick auf Virtualisierungstechnologien und Storage Rechnung zu tragen (vgl. Abbildung 4-2).

Zusammenfassend lässt sich feststellen, dass beim Zugriff auf eine SBC Infrastruktur mehrere verschiedene Antwortzeiten relevant sind:

- Anmelden an der Webseite bzw. am SBC Client
- Abrufen und Darstellen verfügbarer Ressourcen (Anwendungen/Desktops)
- Starten einer Ressource (Anmelden am Server oder Desktop)
- Unmittelbare Interaktion mit der Ressource
- Mittelbare Interaktion mit weiteren Diensten im Backend (z. B. Datenbank-/Dateiserver)
- Beenden der Ressource (Abmelden am Server oder Desktop)

Beim Aufbau und Betrieb einer SBC Infrastruktur sind alle diese Zeiten zu berücksichtigen, um die dauerhafte Benutzerakzeptanz und Kundenzufriedenheit sicherzustellen.

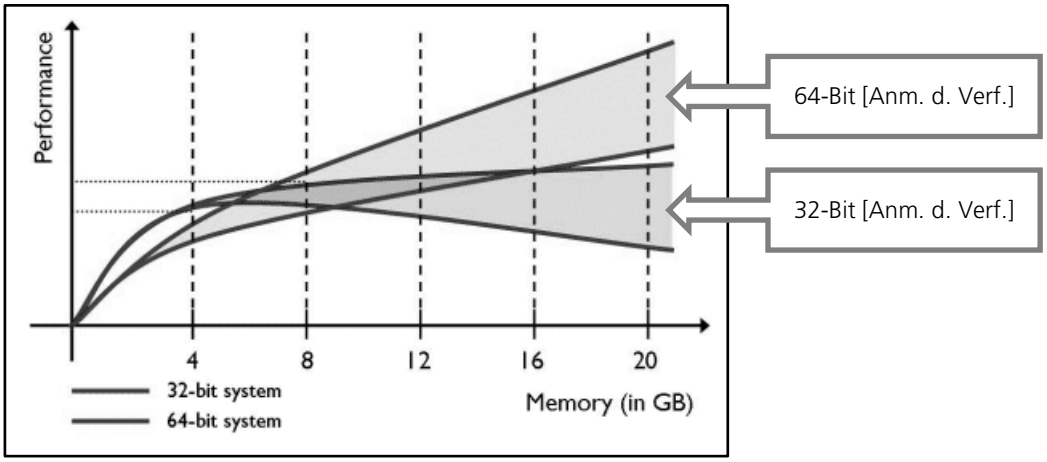
4.1.2 Durchsatz

Der Durchsatz bezeichnet die Rate, mit der ein IT-System Anfragen abarbeitet, gemessen in Operationen pro Zeiteinheit. In [Menascé, 2004], Kap. 1.2.2 werden beispielhaft entsprechende Kennzahlen in Bezug auf ein bestimmtes System bzw. eine bestimmte Ressource genannt. So kann z. B. für eine Webseite die Anzahl der HTTP Requests pro Sekunde oder für eine Festplatte die Anzahl der Input/Output Operationen pro Sekunde (IOPS) untersucht werden. Letztere Kennzahl ist insbesondere bei Planung und Betrieb einer Desktop-Virtualisierung relevant. Im Allgemeinen muss der Durchsatz im Kontext einer SBC Infrastruktur auf verschiedenen Ebenen betrachtet werden. So ist natürlich auch für das Webinterface bzw. den Webservice, der die SBC Clients versorgt, zu betrachten, wie viele HTTP Requests pro Sekunde dieser verarbeiten kann. Für ein Gateway, ebenso wie für einen Terminal Server oder einen Host für virtuelle Desktops, ist zu betrachten, wie viele Benutzersitzungen parallel ausgeführt werden können. Dies führt bei einer granulareren Betrachtung wiederum zur Auslegung des Storage im Hinblick auf die benötigten IOPS sowie zur Dimensionierung von CPU und Hauptspeicher sowohl der physischen als auch der virtuellen Systeme.

Der Durchsatz sollte bereits bei der Planung einer SBC Infrastruktur im Hinlick auf die zu erwartende Last betrachtet werden, da im Fall einer Überlastung – im Englischen als *Thrashing* bezeichnet (vgl. [Menascé, 2004], S. 15) – die Performance eines Systems nicht nur stagnieren, sondern sogar abnehmen kann. Konkret auf den Betrieb von Terminal Servern bezogen sei als Beispiel für *Thrashing* der Vergleich des Einsatzes bestimmter Anwendungsprogramme auf 32-Bit und 64-Bit Betriebssystemen angeführt (vgl. [Hiebel, 2008], S. 84 ff.): „32-Bit Applikationen auf einem 64-Bit Betriebssystem einzusetzen, führt (…) zu einer ineffizienteren Nutzung des Speichers. Microsoft und HP erwarten einen um 50-100 Prozent höheren Speicherbedarf für die 64-Bit Plattform sowie eine bis zu 20 Prozentpunkte höhere Prozessorlast (vgl. [Microsoft, 2005], S. 12-17), da die Prozessoren mit dem Übersetzen von 32-Bit auf 64-Bit

Datenstrukturen zusätzlich belastet werden. Diese Ergebnisse werden gestützt von einem technischen Whitepaper des Terminal Server-Experten (…) Tritsch, der für identische Serverhardware mit weniger als 16 GB unter einem x64 Betriebssystem eine geringere Leistung ermittelte als unter einem x86 Betriebssystem (vgl. [Tritsch, 2007]). Eine weitere Studie zur Skalierbarkeit von typischen Office Programmen weist eine um bis zu 100 Prozent höhere Speicherbelegung für einzelne Programme aus, die in bis zu 50 Prozent höheren Anforderungen für die Benutzersitzungen insgesamt resultiert (vgl. [Tritsch, 2007a], S. 15ff). (…) Aus diesen Kennzahlen ergibt sich, dass x64 Betriebssysteme in der Regel nur auf entsprechend dimensionierter Serverhardware ihre Vorteile ausspielen können. Erst Systeme mit 4-8 Prozessorkernen und mehr als 16 GB Hauptspeicher lassen sich in Verbindung mit einem 64-Bit Betriebssystem effizienter nutzen und erlauben es, eine größere Anzahl an Benutzersitzungen zu unterstützen."

Abbildung 4-3: Skalierbarkeit von 32- und 64-Bit Systemen



Quelle: [Tritsch, 2007]

Entsprechend diesen Ausführungen lässt sich anhand der Ergebnisse aus [Tritsch, 2007] der Effekt des Thrashings am Beispiel der 32-Bit Systeme deutlich erkennen (vgl. Abbildung 4-3): Der Overhead beim Verwalten des Hauptspeichers bedeutet eine zusätzliche Belastung für das System, so dass dessen Performance insgesamt sinkt. Ein ähnliches Verhalten zeigt ein Terminal Server, wenn bei einer fixen Menge an Hauptspeicher die Anzahl an Benutzersitzungen über die verfügbare Kapazität hinaus ansteigt und das System auf Grund von zu geringem physischen Hauptspeicher zum Auslagern (Paging) gezwungen wird. Beim Aufbau und Betrieb einer SBC Infrastruktur ist entsprechend auch und insbesondere der Durchsatz zu berücksichtigen, um die dauerhafte Benutzerakzeptanz und Kundenzufriedenheit sicherzustellen.

4.1.3 Verfügbarkeit/Zuverlässigkeit

Die Verfügbarkeit, angegeben in Prozent, ist der Anteil der Zeit, in dem ein System betriebsbereit und für seine Endanwender bzw. Kunden verfügbar ist. Dabei ist zu berücksichtigen, dass die Ausfallzeit vom zu betrachtenden Zeitraum, d. h. pro Tag, Woche,

Monat oder Jahr, abhängt. Je nach Zeitraum ergeben sich unterschiedliche absolute Werte für die maximale Dauer der Ausfallzeit (vgl. Tabelle 4-1). Es ist weiterhin zu berücksichtigen, dass die maximale Dauer der Ausfallzeit keine Aussage darüber trifft, ob es sich um *einen* Ausfall oder *mehrere* Ausfälle von kürzerer Dauer handelt. So beträgt die Ausfallzeit bei einer Verfügbarkeit von 99,99 Prozent bezogen auf einen Monat beispielsweise 4:19 Minuten. Dies kann bedeuten, dass in diesem Monat bspw. genau ein Ausfall von eben dieser Dauer oder aber zwei Ausfälle mit einer Dauer von 2:00 sowie 2:19 Minuten auftreten könnten.

Tabelle 4-1: Ausfallzeiten pro Tag, Woche, Monat und Jahr

Verfügbarkeit (%)	pro Tag	pro Woche	pro Monat	pro Jahr
99,0000	14:24	1:40:48	7:12	87:36:00
99,9000	1:26	10:05	43:12	8:45:36
99,9900	0:09	1:00	4:19	52:34
99,9990	0:01	0:06	0:26	5:15
99,9999	0:00	0:01	0:03	0:32

Quelle: Eigene Darstellung

Die Begriffe der Verfügbarkeit und der Zuverlässigkeit eines Systems werden in der Praxis oftmals synonym gebraucht. Die Zuverlässigkeit ist jedoch weiter gefasst und „(…) bezeichnet die Fähigkeit eines Systems, während einer vorgegebenen Zeitdauer bei zulässigen Betriebsbedingungen die spezifizierte Funktion zu erbringen“ (vgl. [Echtle, 1990], S. 3). Die Zuverlässigkeit bezieht sich also auf die korrekte Funktion des Systems, über die bloße Verfügbarkeit hinaus – darauf, dass sich das System für den Endanwender erwartungskonform verhält und dessen gewünschte Berechnungen und Abfragen korrekt ausführt.

Die Hauptursachen für Ausfallzeiten sind Fehlerzustände und Überlastung. Ersteren lässt sich im Kontext von SBC Infrastrukturen durch das redundante Auslegen von Systemen vorbeugen. So wurden in Kapitel 2 bereits verschiedene Möglichkeiten diskutiert, Terminal Server innerhalb von Farmen redundant zu betreiben und auch die unterliegenden Virtualisierungshosts mit hoher Verfügbarkeit auszulegen. Eine genauere Systematisierung von Begriffen und Maßnahmen im Hinblick auf Fehlertoleranzverfahren findet sich im folgenden Absatz 4.2.

Überlastung (Thrashing, vgl. vorheriger Absatz 4.1.2) lässt sich zum einen durch eine Limitierung der Zugriffe verhindern – indem beispielsweise ein Webserver so konfiguriert wird, dass er keine neuen TCP Verbindungen mehr annimmt, wenn ein definiertes Maximum erreicht ist, oder ein Terminal Server so konfiguriert wird, dass nur eine bestimmte Anzahl an gleichzeitigen Benutzersitzungen akzeptiert wird.

Eine solche Limitierung würde natürlich bedeuten, dass Anfragen von Kunden abgewiesen werden müssen, was der Akzeptanz seitens der Endanwender abträglich wäre. Entsprechend

empfiehlt es sich, mögliche Zustände der Überlastung möglichst frühzeitig zu erkennen und die Infrastruktur zu skalieren (vgl. Kapitel 4.1.5), so dass die Überlast vermieden werden kann.

4.1.4 Sicherheit

Die Sicherheit umfasst nach Menascé et al. die Vertraulichkeit, Integrität sowie die Nicht-Anfechtbarkeit von Nachrichten und Informationen innerhalb eines IT-Systems. Allgemein dienen Authentifizierungs- und Verschlüsselungsmechanismen der Umsetzung dieser Anforderungen. Bezogen auf Planung und Betrieb von SBC Infrastrukturen wurden die relevanten Techniken und Methoden in Kapitel 3 umfassend erläutert.

Während die zuvor genannte Definition sich nur auf die Sicherheit der mit dem System ausgetauschten und darin verarbeiteten Informationen bezieht, also keine Aussage über mögliche Gefahren für Leib und Leben der Anwender oder deren Hab und Gut trifft, geht Echtle explizit darauf ein. Demnach bezeichnet Sicherheit „(…) das Nicht-Vorhandensein einer Gefahr für Menschen oder Sachwerte (…)“, was insbesondere in Bereichen kritischer Infrastruktur, wie z. B. Verkehrslenkung, Energiegewinnung und -verteilung oder industrieller Fertigung relevant ist (vgl. [Echtle, 1990], S. 2/3) und für den Einsatz von SBC Infrastrukturen in weniger kritischem Umfeld selbstverständlich sein sollte.

4.1.5 Skalierbarkeit

Ein System wird als skalierbar bezeichnet, wenn sich seine Performance mit steigender Anzahl an Benutzern bzw. Last nicht verringert (vgl. [Menascé, 2004], S. 19). Wenn ein System an seine Kapazitätsgrenze stößt, sollte seine Architektur so beschaffen sein, dass es sich durch Hinzufügen weiterer Ressourcen möglichst einfach anpassen lässt. Bezogen auf den Betrieb von Terminal Servern innerhalb einer SBC Infrastruktur kann dies beispielsweise auf zwei Arten realisiert werden:

- **Vertikale Skalierbarkeit („Scale up“):** Einzelne Server werden in ihrer Leistungsfähigkeit vergrößert, z. B. indem den Servern jeweils mehr Prozessoren und/oder mehr Hauptspeicher hinzugefügt wird. Die Anzahl an Systemen bleibt konstant.
- **Horizontale Skalierbarkeit („Scale out“):** Es werden der Umgebung weitere gleichartige Server hinzugefügt. Die Anzahl an Systemen steigt.

Die spätere Skalierbarkeit sollte bereits bei der Planung einer Infrastruktur berücksichtigt werden, so dass es möglichst vermieden werden kann, im Lebenszyklus des Gesamtsystems einen grundlegenden Technologiewechsel vollziehen zu müssen, um steigenden Nutzerzahlen gerecht zu werden.

4.1.6 Erweiterbarkeit

Die Erweiterbarkeit eines Systems zielt im Gegensatz zur Skalierbarkeit nicht auf dessen Kapazität, sondern auf dessen Funktionalität. Wiederum bezogen auf eine SBC Infrastruktur sollte entsprechend bereits in der Planungsphase berücksichtigt werden, dass über den

Lebenszyklus des Gesamtsystems neue Funktionen hinzuzufügen sind, die in der Anfangsphase noch nicht erforderlich sind. So mag es beispielsweise zu Beginn eines Projekts ausreichend sein, Anwendungen basierend auf Terminal Servern bereitzustellen, doch im weiteren Verlauf kommen evtl. Applikationen hinzu, die zwingend virtuelle Desktops benötigen.

4.2 Fehlertoleranzverfahren

Eine SBC Infrastruktur sollte auch beim Auftreten von Fehlern innerhalb der Umgebung weiterhin in der Lage sein, die ihr zugedachten Aufgaben zu erfüllen. Hier findet der Begriff der Fehlertoleranz Verwendung. Diese „(…) bezeichnet die Fähigkeit eines Systems, auch mit einer begrenzten Anzahl fehlerhafter Komponenten seine spezifizierte Funktion zu erfüllen (…)" (vgl. [Echtle, 1990], S. 9).

4.2.1 Redundanz

Um eine Infrastruktur gegen Fehler abzusichern, werden ihre Komponenten typischerweise redundant ausgelegt. Der Begriff der Redundanz „(…) bezeichnet das funktionsbereite Vorhandensein von mehr technischen Mitteln, als für die spezifizierten Nutzfunktionen eines Systems benötigt werden (…)" und lässt sich weiter systematisieren (vgl. [Echtle, 1990], S. 50ff.), wobei im Falle von SBC Infrastrukturen besonders zwei Ausprägungen relevant sind:

- **Strukturelle Redundanz** liegt vor, wenn eine SBC Infrastruktur um für den eigentlichen Betrieb nicht zwingend notwendige Komponenten erweitert wird, also z. B. mehr Virtualisierungshosts oder mehr Terminal Server VM installiert werden als im Regelbetrieb für eine angenomme Anzahl an Anwendern nötig wären.
- **Informationsredundanz** liegt vor, wenn zusätzlich zu den eigentlichen Nutzdaten weitere Informationen gespeichert werden, etwa Prüfsummen auf den Festplatten der in einer Infrastruktur eingesetzten Dateiserver. Da in diesem Fall in der Regel auch zusätzliche Festplatten als sogenannte „Hot Spare" vorgehalten werden, liegt eine Kombination von struktureller Redundanz und Informationsredundanz vor.

Eine weitere Unterscheidung ist möglich anhand der Frage, ob die redundanten Komponenten auch im Regelbetrieb dauerhaft aktiv sind oder erst im Fehlerfall aktiviert werden:

- Ersteres ist mit dem Begriff der **statischen Redundanz** beschrieben und in SBC Infrastrukturen typischerweise der Fall: Sofern mehr Virtualisierungshosts oder Terminal Server eingesetzt werden als nötig, laufen diese im Regelfall dauerhaft aktiv gleichberechtigt im Lastverbund, um die zusätzlich vorgehaltene Hardware auch zur Steigerung der Leistung zu verwenden und nicht ungenutzt zu lassen.
- Demgegenüber findet der Begriff der **dynamischen Redundanz** Verwendung, wenn redundante Ressourcen vorgehalten werden, die erst im Fehlerfall aktiviert werden und im Regelbetrieb nicht im Einsatz sind. Dies ist so beim zuvor genannten Beispiel der „Hot Spare" Festplatten der Fall, die erst beim Ausfall einer der produktiven Festplatten zum Einsatz kommen, um anhand der Prüfsummen Informationen neu zu berechnen. Speziell beim Betrieb von SBC Infrastrukturen können auch dynamisch zur Laufzeit

virtuelle Desktops zusätzlich gestartet oder wieder heruntergefahren werden. Dies ist beispielweise mit der Lösung Citrix XenDesktop (vgl. Kapitel 2.1.3.2) abhängig von der Tageszeit oder auch von der tatsächlichen Anzahl aktiver Benutzer im System möglich. Für komplette Blade Center, Server oder Netzwerkkomponenten verbietet sich diese Art der Redundanz in der Regel aus wirtschaftlichen Überlegungen, sofern es sich nicht um sehr große Infrastrukturen handelt, die hunderte gleichartiger Hardware-Komponenten einsetzen und mehrere Systeme gleichen Typs als sogenannten „Cold Standby" vorhalten.

4.2.2 Fehlerdiagnose

Bezogen auf das Erkennen und Diagnostizieren von Fehlerzuständen können grundsätzlich zwei Arten von Tests unterschieden werden (vgl. [Echtle, 1990], S. 73) – zum einen der **Strukturtest**, der eine Umgebung *von innen* überprüft und dazu tiefere Kenntnis ihrer Struktur und des Zusammenspiels ihrer Komponenten, Dienste sowie deren Funktions-zuordnungen erfordert; zum anderen der **Funktionstest**, welcher ohne weitergehende Informationen über die inneren Zusammenhänge der Infrastruktur *von außen* prüft, ob die angedachte Funktionalität noch erbracht wird.

Im Falle einer SBC Infrastruktur kommen als Werkzeuge für Strukturtests typische Monitoring-Werkzeuge in Frage, welche überprüfen, ob die einzelnen Virtualisierungshosts, Server und Netzwerkkomponenten noch erreichbar sind und notwendige Funktionen und Dienste darauf laufen. Als bekannte Beispiele seien hier Nagios[86], Zabbix[87] oder der SolarWinds ipMonitor[88] erwähnt. Einen Funktionstest für SBC Umgebungen realisiert die Software Login PI[89], welche für die Infrastrukturlösungen der Hersteller Citrix, Microsoft und VMware in der Lage ist, automatisiert Benutzersitzungen anzumelden, darin Applikationen sowie Aktionen auszuführen und auf diese Weise nicht nur festzustellen, ob eine Umgebung grundsätzlich funktioniert, sondern auch zu prüfen, ob innerhalb der Sitzungen gewünschte Performancewerte eingehalten werden.

4.2.3 Fehlerbehandlung

Wenn im Rahmen der Diagnose ein Fehlerzustand erkannt wird, ist – durch manuelles Eingreifen oder vorzugsweise automatisiert – schnellstmöglich dafür Sorge zu tragen, dass dieser Fehlerzustand entsprechend behandelt wird und die bestimmungsgemäße Funktion des Gesamtsystems nicht beeinträchtigt.

Dies kann zunächst durch **Rekonfigurierung** geschehen, indem fehlerhafte Komponenten außer Betrieb gesetzt und die ihnen zugedachten Aufgaben auf andere Komponenten

86 Nagios: https://www.nagios.org (abgerufen am 01.02.2016).
87 Zabbix: http://www.zabbix.com (abgerufen am 01.02.2016).
88 SolarWinds ipMonitor: http://www.solarwinds.com/products/ipmonitor/ (abgerufen am 01.02.2016).
89 Login PI: http://www.loginvsi.com/products/login-pi (abgerufen am 01.02.2016).

verlagert werden. Die „**Rekonfigurierung** ordnet jeder Komponente eines betrachteten Subsystems fehlerfreie Komponenten zu, um die benötigten Funktionen zu erbringen. Meist handelt es sich bei dem ‚betrachteten Subsystem' um eine Schicht, so daß Rekonfigurierung einen Vorgang an der Grenze zur nächstniedrigeren Schicht darstellt. (…)" (vgl. [Echtle, 1990], S. 86)

Im Hinblick auf den Betrieb von SBC Infrastrukturen kann dies direkt auf die fünf Schichten des in Kapitel 5.4.1 entworfenen Referenzmodells angewendet werden. Auch bezogen auf dieses Modell betrifft die Rekonfigurierung meist eine Schicht. So können beispielsweise auf der Zugriffsschicht jeweils mehrere paketfilternde Firewalls, Gateways oder Web-Server mittels struktureller Redundanz hochverfügbar ausgelegt werden, so dass einzelne Komponenten im Fehlerfall abgeschaltet werden können.

Auch auf der Anwendungsschicht werden in der Regel mehrere gleichartige Terminal Server oder virtuelle Desktops parallel betrieben. Sollten in einzelnen Systemen Fehler auftreten, können diese einfach für Anmeldungen gesperrt oder ganz heruntergefahren werden. Gleiches gilt für die physischen Virtualisierungshosts auf der Infrastrukturschicht. Soweit es noch möglich und die Kapazität der verbleibenden System ausreichend ist, können die virtuellen Maschinen von einem oder mehreren fehlerhaften Hosts im laufenden Betrieb auf andere Hosts migriert und die betroffenen Systeme dann ausgeschaltet werden. Analoge Überlegungen lassen sich für sämtliche Hilfsdienste in der Backend-Schicht anstellen.

Unabhängig davon, ob sich ein Fehlerzustand kurzfristig durch Rekonfigurierung vor den Endanwendern verbergen lässt, ist der Fehler zu beheben, um die fehlerhaften Komponenten wieder in einen ordnungsgemäßen Betriebszustand zu überführen. Die möglichen Maßnahmen hierzu lassen sich folgendermaßen systematisieren:

- Die „**Rückwärtsbehebung** versucht, auf einen noch fehlerfreien Zustand in der Vergangenheit zuzugreifen. (…) Rückwärtsbehebung (…) versetzt Komponenten in einen Zustand, den sie bereits in der Vergangenheit angenommen hatten oder als konsistenten Zustand hätten annehmen können." (vgl. [Echtle, 1990], S. 109/110) Dies setzt natürlich voraus, dass entsprechende *Rücksetzpunkte* vorhanden sind, aus denen der ursprüngliche Zustand wiederhergestellt werden kann. Bezogen auf SBC Infrastrukturen kann es sich dabei beispielsweise um die *Wiederherstellungspunkte* handeln, welche sich mit den Bordmitteln aktueller Versionen des Betriebssystems Microsoft Windows erzeugen lassen. Alternativ dazu ist die Wiederherstellung kompletter virtueller Maschinen aus *Snapshots* denkbar, welche periodisch auf der Schicht der Virtualisierungshosts erzeugt werden können. Besonders einfach ist die Wiederherstellung beim Einsatz von Provisionierung (vgl. Kapitel 2.3.1). In diesem Fall kommen *schreibgeschützte Images* zum Einsatz, so dass in der Regel ein Neustart fehlerhafter Systeme ausreicht, um auf einen fehlerfreien Zustand zurückzukehren. Für alle übrigen Dienste in der Backend-Schicht kann herkömmliche *Backup-Software* eingesetzt werden, um Rücksetzpunkte für Betriebssysteme, Active Directory, Datenbanken und weitere Dienste zu erzeugen.

- Demgegenüber versetzt die **Vorwärtsbehebung** „(…) Komponenten in einen konsistenten Zustand, ohne auf Zustandsinformation zurückzugreifen, die in der Vergangenheit zum Zweck der Fehlertoleranz (d. h. als *Rücksetzpunkt*) abgespeichert wurde. (…) Da die Kenntnis der Vergangenheit der Vorwärtsbehebung nicht zur Verfügung steht, ist der konsistente Zustand anwendungsabhängig zu finden." (vgl. [Echtle, 1990], S. 164). Im Hinblick auf SBC Infrastrukturen würde dies z. B. bedeuten, fehlerhafte Betriebssysteme sowie Anwendungen und Dienste von Grund auf neu zu installieren und zu konfigurieren bzw. fehlerhafte Hardware-Komponenten durch neue zu ersetzen und ebenfalls neu zu konfigurieren.
- Einen Sonderfall stellt die dritte Variante der **Seitwärtsbehebung** dar. Diese sieht vor, dass passend zu den produktiven Prozessen innerhalb eines Systems „(...) passive Ersatzprozessexemplare schon seit Betriebsbeginn vorhanden sind, (…) Prinzipiell kommen dafür die bereits erläuterten Techniken der Rückwärts- oder Vorwärtsbehebung in Betracht (…) Dieser Zusatzaufwand läßt sich aber einsparen, wenn mehrere Prozeßexemplare die gleichen Berechnungen ausführen und daher gegenseitig verwendbare Zustandsinformation besitzen (…) so daß sich die Unterscheidung von Rückwärts- und Vorwärtsbehebung erübrigt." (vgl. [Echtle, 1990], S. 215). Bezogen auf SBC Infrastrukturen bedeutet dies in der Regel den Einsatz von Drittanbieter-Produkten, welche Berechnungen einzelner Prozesse, Anwendungen oder virtuelle Maschinen zwischen verschiedenen Virtualisierungshosts oder gar komplett geografisch getrennten Rechenzentren replizieren. Auf Grund der damit verbundenen signifikant höheren Kosten kommen solche Lösungen jedoch typischerweise nur bei unternehmenskritischen Anwendungen mit sehr hohen Anforderungen an die Verfügbarkeit zum Einsatz. Als Beispiele seien die Veeam Availability Suite[90] zur Absicherung von VMware vSphere und Microsoft Hyper-V Virtualisierungshosts sowie Stratus everRun[91] zur Absicherung von Anwendungen und Diensten unter Microsoft Windows und Linux erwähnt.

Fehlertoleranzverfahren mit entsprechenden Redundanzen sowie Maßnahmen zur Fehlerdiagnose und -behandlung sind bereits in den frühen Phasen des im folgenden Kapitel dargestellten Lebenszyklus bei der Planung von SBC Infrastrukturen zu berücksichtigen, um den dauerhaften Betrieb zu gewährleisten.

4.3 Lebenszyklus

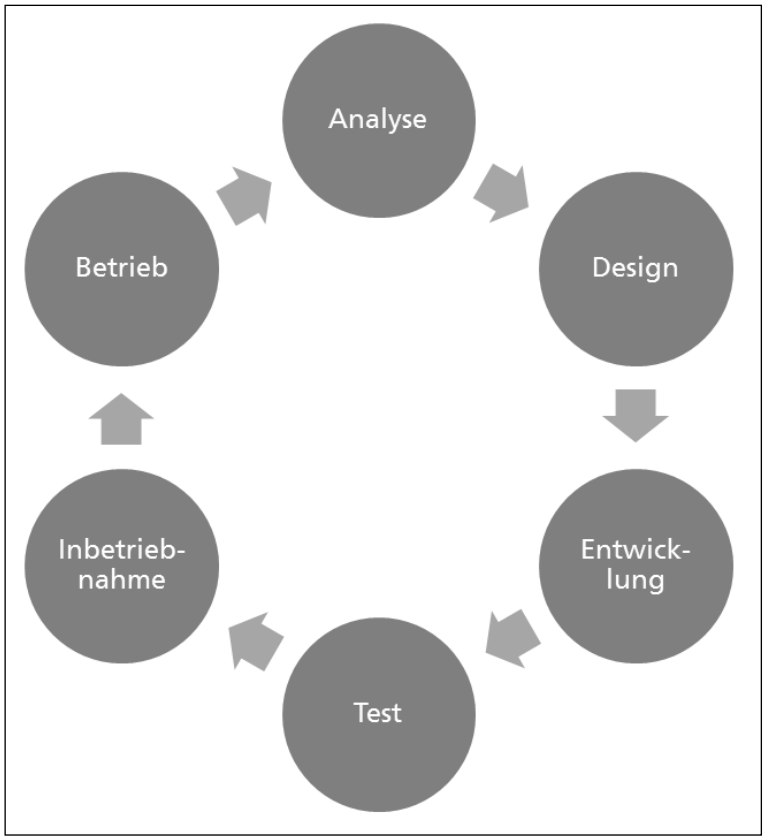
Da die Suche nach den Ursachen für Performance-Engpässe erst im Nachgang eines Projektes ein zeit- und kostenintensives Unterfangen werden kann, sollten Überlegungen bezüglich der zu erwartenden bzw. von einem System geforderten Kapazität bereits zu Beginn des Lebenszyklus des jeweiligen Systems berücksichtigt werden. Nach Menascé et. al. kann der Lebenszyklus in sieben Phasen untergliedert werden (vgl. [Menascé, 2004], S. 20 ff.):

[90] Veeam Availability Suite: https://www.veeam.com/de/data-center-availability-suite.html (abgerufen am 01.02.2016).
[91] Stratus everRun: http://www.stratus.com/solutions/software/everrun-enterprise-express/ (abgerufen am 01.02.2016).

- Analyse und Spezifikation der Anforderungen
- System Design
- System Entwicklung
- Testphase
- Inbetriebnahme
- Betrieb
- Weiterentwicklung

Die Weiterentwicklung im Sinne der Migration eines bestehenden Systems auf ein nachfolgendes kann auch als Wiederholung der ersten Phasen verstanden werden. Als Basis für die Weiterentwicklung dienen dabei die aus dem Betrieb gewonnenen Erfahrungen und ggf. neue oder veränderte Anforderungen (vgl. Abbildung 4-4).

Abbildung 4-4 Weiterentwicklung als kontinuierlicher Prozess



Quelle: Eigene Darstellung

4.3.1 Analyse und Spezifikation der Anforderungen

Gemeinsam mit den Kunden bzw. Endanwendern des zu schaffenden IT-Systems sind die Anforderungen an seine Funktionalität und Leistung zu definieren. Funktionale Anforderungen legen dabei fest, *was* das System leisten soll, d. h. welche Funktionen es bieten soll und welche Eingaben nötig sind sowie welche Ausgaben erwartet werden. Die funktionalen

Anforderungen können bereits technische Spezifikationen beinhalten und beispielsweise im Falle einer SBC Infrastruktur festlegen, dass Applikationen basierend auf Microsoft Windows Terminal Servern unter Windows Server 2008 R2 bereitgestellt werden sollen.

Nicht-funktionale Anforderungen beschreiben demgegenüber, *wie* ein System seine Funktionalität erbringen soll, d. h. welche QoS Anforderungen erfüllt sein müssen. Dies betrifft die Kapazität, Verfügbarkeit, Sicherheit sowie Definition von Lasten, die das System bewältigen soll. Für Terminal Server kann dies beispielsweise bedeuten, dass gefordert wird, dass ein Gesamtsystem in der Spitze mindestens 25 Sitzungen pro Server bei einem gegebenen Nutzungsprofil pro Sitzung bedienen kann.

4.3.2 System Design

Im Hinblick auf die spätere Kapazität des Gesamtsystems ist dies die wichtigste Phase, in deren Verlauf zu klären ist, wie das System in die Lage versetzt werden soll, die zuvor festgelegten Anforderungen zu erfüllen. Dies umfasst die Definition und Auswahl der zu verwendenden Komponenten sowie die Definition von Schnittstellen zwischen den einzelnen Systembestandteilen. Der Fokus sollte beim Design insbesondere auf den nicht-funktionalen Anforderungen liegen.

4.3.3 System Entwicklung

In dieser Phase wird das zuvor konzipierte System implementiert. Für Projekte im Bereich der Software-Entwicklung kann dies bedeuten, dass bereits vorhandene Komponenten angepasst und weiterverwendet oder komplett neue Code-Bestandteile geschrieben werden. Bezogen auf SBC Infrastrukturen, die dem Bereich der IT-Systemtechnik zuzuordnen sind, umfasst die Entwicklung und Implementierung maßgeblich das Zusammenfügen und Vernetzen der einzelnen Hard- und Softwarebestandteile und nur zu einem geringen Anteil die Implementierung eigener Programmteile – dies nur, soweit beispielsweise Webservices und interaktive Webseiten für die Anmeldung am System individuell angepasst werden müssen.

Unabhängig von der Natur des Projekts sollten bereits in dieser Phase Schnittstellen für das spätere Monitoring und die Erfassung statistischer Daten zur Auslastung des Systems vorgesehen werden, da entsprechende Daten zur Überwachung von QoS Anforderungen und als Input für die Weiterentwicklung des Systems benötigt werden.

4.3.4 Testphase

In der Testphase werden zunächst im Rahmen eines Modultests die einzelnen Komponenten getestet und schließlich zu einem Gesamtsystem zusammengefügt, das dann als Ganzes einem Systemtest unterzogen wird. Neben der Erfüllung der funktionalen Anforderungen sollte dabei insbesondere ein Schwerpunkt auch auf die nicht-funktionalen Anforderungen gelegt werden. Für eine SBC Infrastruktur kann dies in Form eines Lasttests geschehen, in dessen Verlauf automatisiert durch entsprechende Softwarelösungen zahlreiche virtuelle Benutzersitzungen

angemeldet werden, die sodann skriptbasiert typische Aufgaben der späteren Benutzer des Systems simulieren (vgl. Kapitel 5.4.4.1).

4.3.5 Inbetriebnahme

Nachdem das System, ggf. in kleinerem Umfang und unter kontrollierten Bedingungen, getestet worden ist, wird es zur allgemeinen Nutzung freigegeben. Dabei ist es gemäß der zu erwartenden Last und gewünschten Kapazität zu konfigurieren, was im Falle einer SBC Infrastruktur z. B. die Anzahl gleichzeitig laufender Terminal Server oder virtueller Desktops sowie deren Ausstattung mit Hauptspeicher und (virtuellen) Prozessoren umfasst.

4.3.6 Betrieb

In der Betriebsphase werden die zuvor implementierten Schnittstellen für das Monitoring und die Erfassung statistischer Daten zur Auslastung des Systems kontinuierlich genutzt, um das System proaktiv im Hinblick auf technische Defekte und anderweitige Ausfälle zu überwachen. Des Weiteren kann auf diese Weise sichergestellt werden, dass die zugesicherten QoS Anforderungen eingehalten werden. Zu überwachende Parameter in Bezug auf SBC Infrastrukturen umfassen beispielsweise:

- **Auslastung:** Wie viele Benutzer sind zu welcher Zeit typischerweise im System aktiv? Welche Anwendungen nutzen diese, d. h. lassen sich Benutzer zu Gruppen mit ähnlichem Anforderungsprofil clustern (vgl. Kapitel 6.5.4)?
- **Externe Performance:** Dies umfasst sämtliche Parameter, die von den Endbenutzern selbst wahrgenommen werden, so z. B. die Ladezeit des Webinterfaces, Anmeldezeit am Terminal Server oder virtuellen Desktop oder das allgemeine Antwortverhalten der Remote-Sitzung.
- **Interne Performance:** Dies beinhaltet sämtliche Faktoren, die von den Anwendern nicht unmittelbar wahrgenommen werden, also u. a. Auslastung von Hauptspeicher und Prozessoren, Netzwerkverkehr oder auch die IOPS im Storage-Subsystem.
- **Verfügbarkeit:** Die Verfügbarkeit kann auf verschiedenen Ebenen überwacht und erfasst werden, so beispielsweise innerhalb der SBC Infrastruktur selbst, indem einzelne Server, Komponenten und Dienste überwacht und ggf. neu gestartet werden, oder von externen Agenten, d. h. automatisierten Programmen, die das Verhalten eines Endanwenders simulieren und eine virtuelle Benutzersitzung anmelden, um die ordnungsgemäße Funktion des Gesamtsystems zu überprüfen (vgl. Kapitel 4.2).

Die Ergebnisse des Monitorings können zur Optimierung des System herangezogen werden, um z. B. Tageszeit-abhängig mehr oder weniger virtuelle Maschinen zu starten, die Benutzersitzungen annehmen.

4.3.7 Weiterentwicklung

Die Weiterentwicklung des Systems umfasst zum einen die Anpassung im Hinblick auf sich verändernde Last (Skalierbarkeit, Kapitel 4.1.5) sowie auch in Bezug auf sich verändernde

funktionale Anforderungen (Erweiterbarkeit, Kapitel 4.1.6). Letzterer Aspekt berührt dabei natürlich auch unmittelbar Fragen der Skalierbarkeit sowie der Kapazität insgesamt, wenn beispielsweise in einer SBC Infrastruktur, die bislang ausschließlich auf Terminal Servern basierte, neue Anforderungen an die bereitzustellenden Applikationen entweder den zusätzlichen Einsatz von oder den Wechsel auf virtuelle Desktops erfordern.

Im Falle einer bereits etablierten Infrastruktur beginnt der Lebenszyklus also wiederum mit der Analyse, wobei in die neuerliche Erhebung der funktionalen und nicht-funktionalen Anforderungen zusätzlich die gewonnen Erfahrungen aus dem bisherigen Betrieb einfließen.

Das in Kapitel 6 beschriebene Szenario, welche als Beispiel für die Implementierung einer SBC Infrastruktur für kleine bis mittelständische Unternehmen dient, soll als Rahmen für die Konkretisierung und Validierung des im folgenden Kapitel entworfenen Referenzmodells dienen.

5 Entwicklung eines Referenzmodells für SBC Infrastrukturen

Dieses Kapitel beschreibt die Entwicklung eines hersteller- und produktneutralen Referenzmodells für SBC Infrastrukturen, welches dem Ziel dient, die Planung und Weiterentwicklung dieser Infrastrukturen in Hinsicht auf das Kapazitätsmanagement zu unterstützen.

5.1 Der Begriff des Referenzmodells

Die Referenzmodellierung kann verstanden werden als „(…) die Menge aller Handlungen, welche die Konstruktion und Anwendung wiederverwendbarer Modelle (Referenzmodelle) beabsichtigen" (vgl. [Fettke, 2004], S. 7). Neben der möglichen oder tatsächlichen Wiederverwendung stellen Fettke und Loos weiterhin auf den Nutzwert für einen Anwender des Referenzmodells ab. Ein Referenzmodell soll seine Nutzer beim Konstruieren von daraus entwickelten Anwendungsmodellen unterstützen. Daraus leiten die Autoren als zentrale Forderungen an ein solches Modell die Eigenschaften der „*Allgemeingültigkeit*, *Anpassbarkeit* und *Anwendbarkeit*" (vgl. [Fettke, 2004], S. 8) ab, schränken dies jedoch selbst insoweit ein, als dass die Allgemeingültigkeit keinen allumfassenden Anspruch erfüllen kann, da diese objektiv kaum feststellbar ist, sondern nur im Kontext des jeweiligen Anwenders subjektiv bestimmt werden kann. Dies gilt ebenso für die Frage, ob das Modell für einen bestimmten Zweck Empfehlungscharakter aufweist.

Eine kritische Diskussion des Begriffs, welche diesen Gedanken aufgreift, findet sich in [Thomas, 2006]. Auch hier grenzt der Autor den Aspekt der Allgemeingültigkeit dahingehend ein, dass dieser „(…) nicht im Sinne eines Absolutheitsanspruchs des Modells, d. h. eines Anspruchs auf universelle Gültigkeit, zu verstehen ist. Ein Referenzmodell kann lediglich in Bezug auf eine Klasse von Anwendungsfällen, z. B. eine Klasse von Unternehmen oder eine Klasse von Projekten, (allgemein-) gültig sein" (vgl. [Thomas, 2006], S. 12). Der Bezug zu einer konkreten Problemstellung muss also gegeben sein und die Frage, ob sich das Modell als Empfehlung, also als *Referenz*, eignet, ist nur subjektiv aus dem Kontext der Problemstellung heraus bestimmbar.

Thomas unterscheidet weiterhin zwei Perspektiven, nämlich die des Erstellers, der ein Referenzmodell *deklariert*, sowie die des Nutzers, der ein Referenzmodell *akzeptiert* und faktisch verwendet. Vor dem Hintergrund dieser Perspektiven ergeben sich drei zu unterscheidende Fälle (vgl. [Thomas, 2006], S. 14ff.):

1. Ein Referenzmodell wird zwar von seinem Ersteller als solches deklariert, findet jedoch keine Anwender, von denen es benutzt und somit akzeptiert würde.

2. Ein Modell wird von seinen Erstellern nicht explizit als Referenzmodell deklariert, findet aber trotzdem Anwender, die es als solches verstehen, entsprechend verwenden und somit akzeptieren.

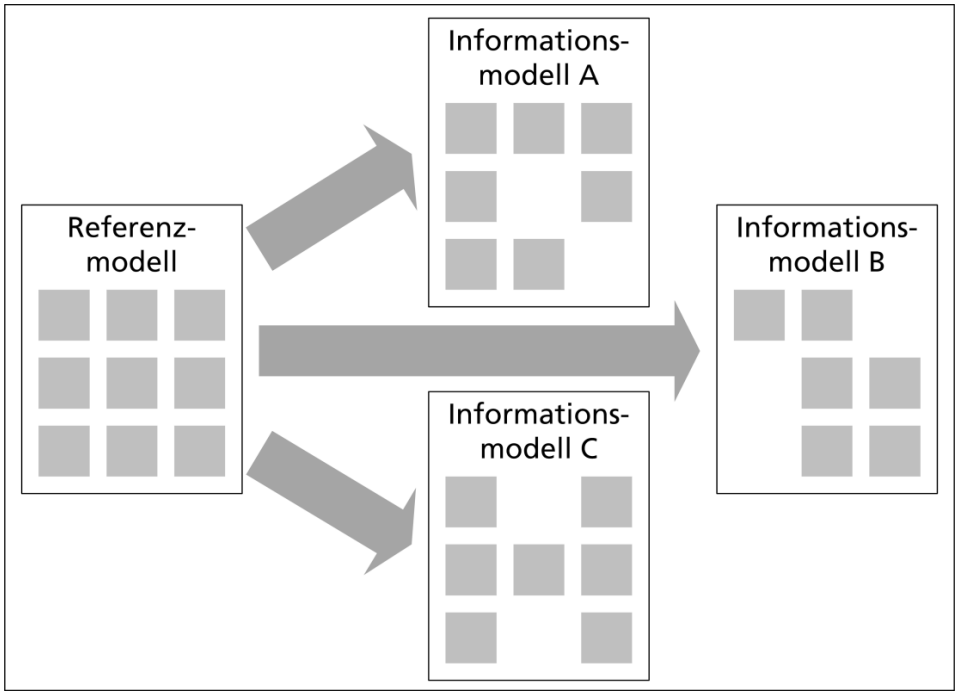
3. Ein Referenzmodell wird von seinen Konstrukteuren als solches deklariert und findet Anwender, die es benutzen und dadurch akzeptieren.

Im Rahmen dieser Arbeit wird der dritte – idealtypische – Fall angenommen, dass nämlich das zu entwickelnde Referenzmodell für SBC Infrastrukturen von Grund auf als solches intendiert ist und seine Anwendbarkeit anhand eines konkreten Szenarios validiert wird. Die Entwicklung folgt damit Thomas' Plädoyer „(…) für einen *nutzungsorientierten* Referenzmodellbegriff. Jedes Modell bzw. Teilmodell, das zu Unterstützung der Konstruktion eines anderen Modells genutzt wird, kann (…) als Referenzmodell angesehen werden" (vgl. [Thomas, 2006], S. 17).

Erst durch die erfolgreiche Nutzung wird also die Absicht des Konstrukteurs bestätigt und das Referenzmodell dadurch akzeptiert. Basierend auf der Akzeptanz der Nutzer als bestimmender Eigenschaft findet sich zum Begriff des Referenzmodells entsprechend die folgende Explikation (vgl. [Thomas, 2006], S. 16):

> *„Ein **Referenzmodell** – ausführlich: Referenzinformationsmodell – ist ein Informationsmodell, das zur Unterstützung der Konstruktion von anderen Modellen genutzt wird."*

Abbildung 5-1: Referenz- und Informationsmodelle



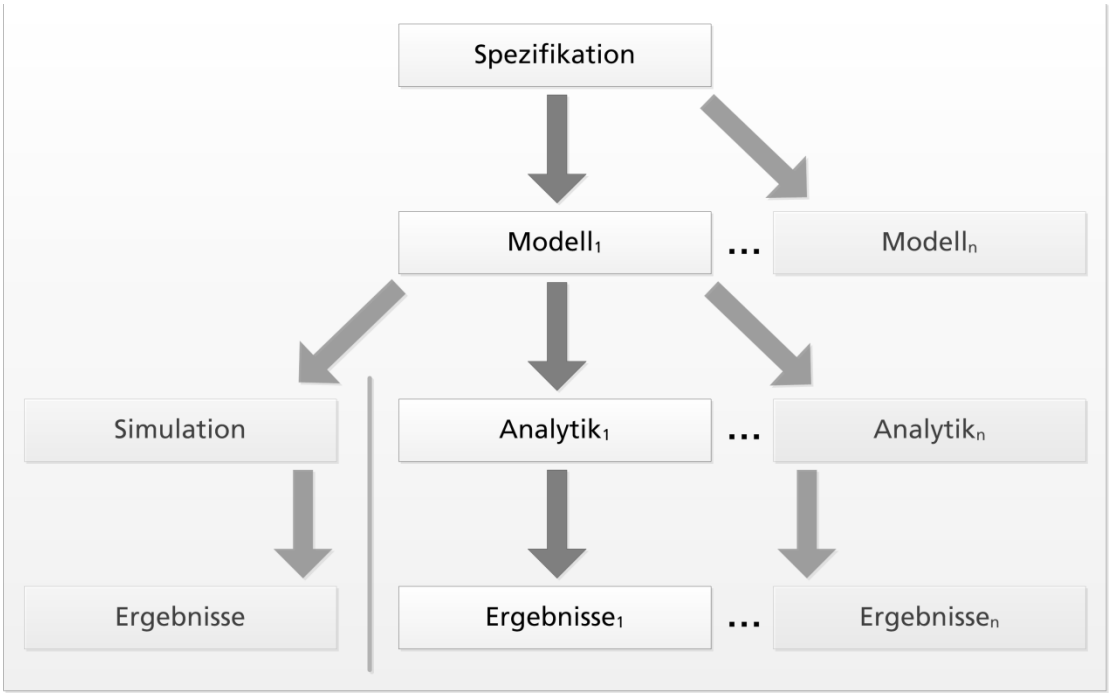
Quelle: Eigene Darstellung

Diese anderen Modelle sind dabei Konkretisierungen des Referenzmodells im Hinblick auf einen bestimmten Anwendungsfall, wobei die Wiederverwendung des Referenzmodells im einfachsten Fall durch simples manuelles Kopieren geschieht. Die Wiederverwendung kann jedoch auch mittels Vererbung bzw. Delegation erfolgen, was als Analogie zur Vererbung in

einer Hierarchie von Klassen bzw. der Instanziierung konkreter Objekte aus einer Klasse in der objektorientierten Programmierung verstanden werden kann (vgl. [Fettke, 2004], S. 15).

Ein aus dem Referenzmodell abgeleitetes Informationsmodell verwendet dabei (nicht zwingend alle) Elemente des ursprünglichen Referenzmodells weiter und konkretisiert oder verändert diese (Abbildung 5-1).

Abbildung 5-2: Transformation des Referenzmodells



Quelle: Eigene Darstellung

Indem das Referenzmodell einen Rahmen für den Planungs- und Konstruktionsprozess darauf aufbauender Informationsmodelle und wiederum daraus entwickelter realweltlicher Lösungen schafft, vereinfacht es die Arbeit der mit dieser Planung und Konstruktion beschäftigten Personen.

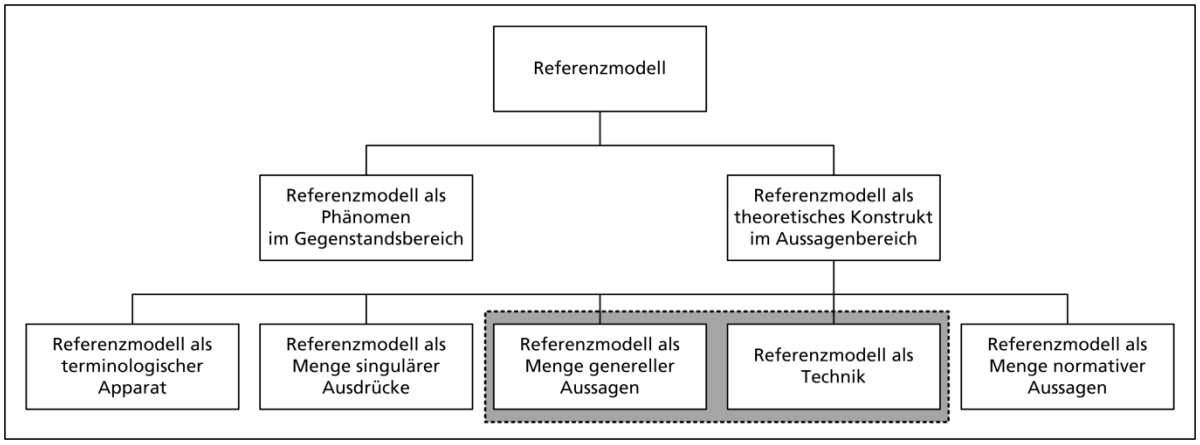
„Die Möglichkeit, sich an den fachlichen Inhalten solcher Referenzmodelle orientieren zu können, verspricht den Modellanwendern einerseits die Einsparung von Zeit und Kosten. Andererseits kann durch die Verwendung eines Referenzmodells die Qualität des zu konstruierenden Modells erhöht werden“ (vgl. [Thomas, 2006], S. 7). Eines oder mehrere der aus der Spezifikation des Referenzmodells entwickelten Informationsmodelle können dann als Vorlage für eine analytische Untersuchung oder die konkrete Implementierung eines Prototypen und die Simulation von Arbeitslasten darauf dienen (Abbildung 5-2 und weiter Kapitel 5.4.4).

5.2 Einordnung des Begriffs in den Kontext

Zur Beantwortung der Frage, wie nun die Referenzmodellierung im Kontext dieser Arbeit die Planung und Weiterentwicklung von SBC Infrastrukturen unterstützen kann, sei zunächst der Begriff des Referenzmodells weiter spezifiziert. Den Rahmen hierfür bildet der Vorschlag zur Systematisierung des Begriffs von Fettke und Loos (vgl. [Fettke, 2004], S. 9), der an dieser Stelle jedoch nicht in seiner Gesamtheit wiedergegeben werden soll.

Für das im Folgenden zu entwickelnde Referenzmodell wird die Information als ausreichend angesehen, dass Referenzmodelle als theoretische Konstrukte interpretiert werden können, die im Aussagenbereich der Wirtschaftsinformatik verortet sind (Abbildung 5-3). Bei einer weiteren Unterscheidung sind insbesondere die Deutungen der Referenzmodelle *als Menge genereller Aussagen* sowie *als Technik* relevant (vgl. [Fettke, 2004], S. 10).

Abbildung 5-3: Systematisierung des Referenzmodellbegriffs



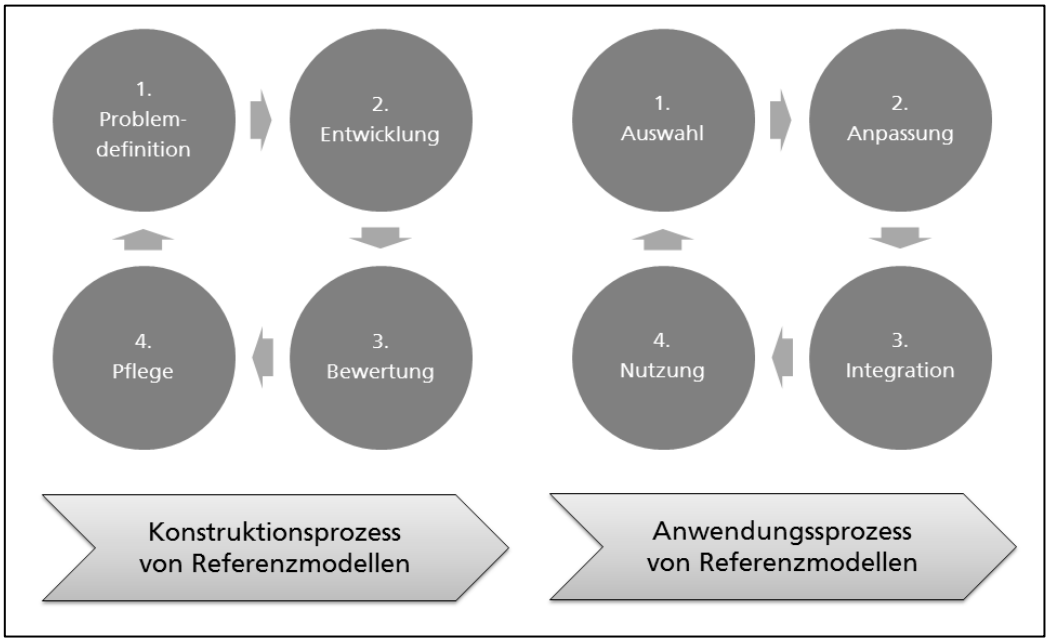
Quelle: Eigene Darstellung (nach [Fettke, 2004], S. 10)

Bezogen auf den intendierten Zweck der Planung und Fortentwicklung von SBC Infrastrukturen, lässt sich der erste Punkt dahingehend interpretieren, dass das zu schaffende Referenzmodell nicht *eine konkrete* SBC Infrastruktur beschreiben soll, sondern eine *Klasse* von SBC Infrastrukturen. Im Hinblick auf eine weitreichende Allgemeingültigkeit sollte die Menge der Infrastrukturen, die dieser Klasse entsprechen, möglichst groß sein und natürlich sollten alle Aussagen, welche das Referenzmodell bezüglich der Klasse von Infrastrukturen trifft, wahr sein.

Der zweite Punkt bezieht sich darauf, dass das Referenzmodell eine Technik mit Relevanz für die Praxis darstellen soll, welche die Arbeit von Informatikern und IT-Systemtechnikern vereinfacht und „ (…) die Gestaltung von Informationssystemen [in diesem Fall also von SBC Infrastrukturen – Anm. d. Verf.] hinsichtlich Kosten-, Zeit-, Qualitäts-, Risiko- oder Wettbewerbssituationsmaßgrößen verbessert“ (vgl. [Fettke, 2004], S. 11 und [Becker, 2003], S. 416).

Fettke und Loos untergliedern den Prozess der Referenzmodellierung in zwei Phasen (vgl. [Fettke, 2005], S. 22/23), nämlich die Konstruktion durch den oder die Ersteller des Modells sowie die darauf folgende Nutzung des Modells durch dessen Anwender (Abbildung 5-4).

Abbildung 5-4: Phasen im Prozess der Referenzmodellierung



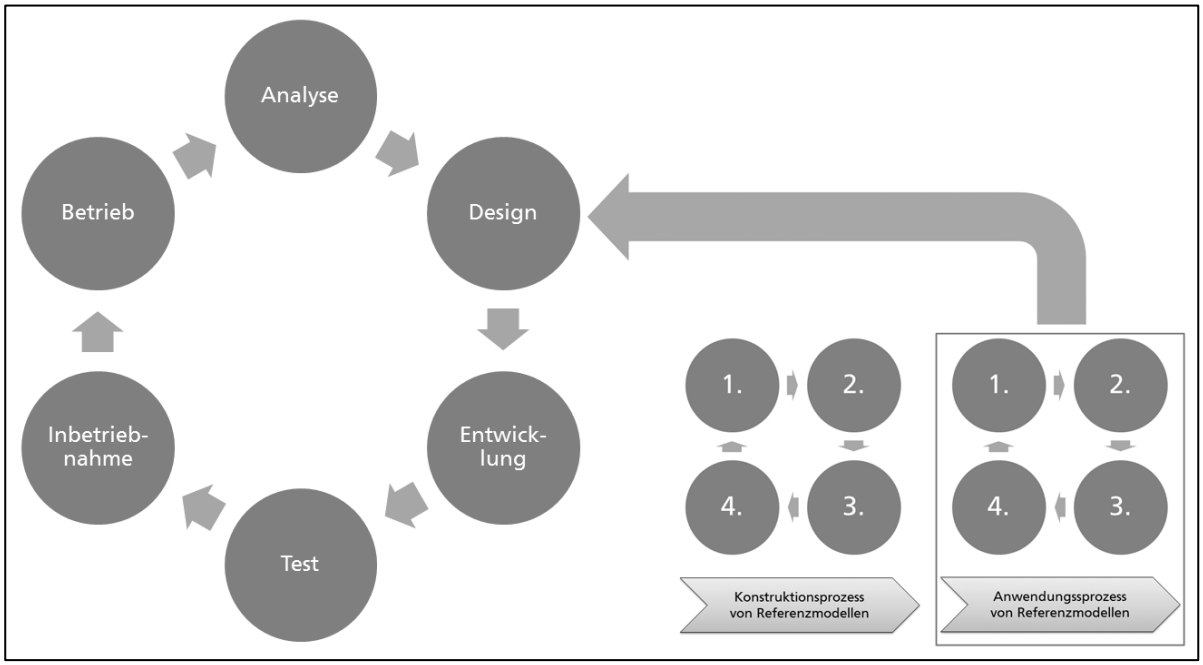
Quelle: Eigene Darstellung (nach [Fettke, 2005], S. 22)

Im *Konstruktionsprozess* ist die Problemdefinition (**1**) im vorliegenden Fall gegeben durch die in den Kapiteln 2 und 3 in generischer Form beschriebenen SBC Infrastrukturen, deren Planung und Weiterentwicklung zu unterstützen ist. Für solche Umgebungen ist eine modellhafte Lösung zu entwickeln (**2**), deren Abstraktionsgrad so gestaltet ist, dass die Klasse der von dem Referenzmodell erfassten Anwendungsfälle möglichst groß ist, diese aber so genau beschreibt, dass daraus ein Nutzwert für die Betreiber der Umgebungen entsteht.

Inwieweit dies der Fall ist, ist – soweit möglich – bereits während der Konstruktion zu bewerten (**3**), mindestens aber als Rückkopplung aus dem Anwendungsprozess nachträglich zu validieren. Das Referenzmodell wird daraufhin, auch im Rahmen sich verändernder technischer Randbedingungen, gepflegt (**4**) und soweit nötig angepasst.

Der *Anwendungsprozess* umfasst die Handlungsschritte, welche ein Nutzer des Referenzmodells vollzieht. Die Auswahl (**1**) ist im vorliegenden Fall bereits vorgegeben durch das in Kapitel 5.4 beschriebene Referenzmodell. Im Rahmen der Anpassung (**2**) wird das Referenzmodell soweit konkretisiert, dass es für einen bestimmten Anwendungsfall verwendbar ist (vgl. Kapitel 5.4.2). Eine Integration (**3**) ist nur erforderlich, soweit mehr als ein Referenzmodell zum Einsatz kommen soll, was im Rahmen dieser Arbeit jedoch nicht gegeben ist. Es folgt somit schließlich die Nutzung (**4**) des angepassten Referenzmodells zur Erreichung der im Vorhinein gesetzten Ziele (vgl. Kapitel 6).

Abbildung 5-5: Referenzmodell – Integration in den Lebenszyklus



Quelle: Eigene Darstellung

Der Anwendungsprozess des Referenzmodells stellt damit ein Hilfsmittel dar, welches in der Design-Phase des Lebenszyklus (vgl. Kapitel 4.3) zum Einsatz kommt (Abbildung 5-5). Damit ein Referenzmodell an dieser Stelle von Nutzen sein kann, ist es inhaltlich und methodisch geeignet zu hinterlegen. Fettke und Loos sprechen in diesem Zusammenhang vom *Kontext der Referenzmodellierung* und verwenden die Begriffe der *Referenzmodellierungssprachen* und *-methoden* (vgl. [Fettke, 2004], S. 12).

Als Beispiele für mögliche *Referenzmodellierungssprachen* seien die Ereignisgesteuerte Prozesskette (EPK), Entity Relationship Modelle (ERM), die Specification and Description Language (SDL), die Unified Modeling Language (UML) oder die Extensible Markup Language (XML) angeführt. Den Rahmen für die Verwendung der Sprache bildet jeweils eine *Referenzmodellierungsmethode*, welche dem Anwender die erforderlichen Handlungsschritte empfiehlt, um aus den Sprachelementen ein Modell zu entwickeln.

Wiederum dafür bildet der *Kontext* den realweltlichen Bezugsrahmen, welcher den Autoren zu Folge durch „psychologische, soziale, organisatorische, technische, wirtschaftliche und andere Faktoren“ die Modellwelt beeinflusst (vgl. [Fettke, 2004], S. 12). So gibt der Kontext der SBC Infrastrukturen, die im Bereich der IT-Systemtechnik verortet sind, beispielsweise vor, dass Administratoren und IT-Systemtechniker die Zielgruppe bilden und entsprechend Methode und Sprache so zu wählen bzw. gestalten sind, dass sie von der Zielgruppe akzeptiert, verstanden und angewendet werden.

5.3 Referenzmodelle und -modellierungssprachen

Beispiele für die Verwendung von Ansätzen der Referenzmodellierung finden sich in verschiedenen Bereichen und Disziplinen der Wirtschaftswissenschaften und Wirschaftsinformatik.

So können Referenzmodelle bei der Unternehmensplanung und dem Entwickeln von Geschäftsprozessen unterstützen. In diesem Fall, der Verwendung von Referenzmodellen im Hinblick auf das Business Engineering, stehen nicht unbedingt konkrete technische (Software-) Lösungen im Vordergrund, sondern die Strategie eines Unternehmens sowie dessen interne Aufbau- und Ablauforganisation (vgl. [Fettke, 2005]).

Strategie und Prozesse haben jedoch wiederum Auswirkungen auf das zu Grunde liegende Softwaresystem und beziehen dieses mit ein. Ein Beispiel hierfür ist das Referenzmodell der Branchenlösung SAP R/3, welches die Prozessgestaltung im Unternehmen, aber auch die konkrete Implementierung der unterliegenden Standardsoftware und ihrer Prozesse umfasst (vgl. [Müller-C., 2001], S. VI-5ff.).

Demgegenüber ist ein prominentes Beispiel für die Verwendung von Referenzmodellen mit rein technischem Hintergrund das in den Kapiteln 2 und 3 bereits mehrfach erwähnte ISO/OSI-Referenzmodell, das auch als ISO/OSI-7-Schichten-Modell (vgl. [ISO, 1994]) oder kurz OSI-Modell bekannt ist. Dieses Modell dient in der Informatik und IT-Systemtechnik dazu, die Kommunikationsbeziehungen zwischen verschiedenen Systemen zu beschreiben sowie deren Planung und Weiterentwicklung zu unterstützen. Das Referenzmodell differenziert dazu sieben aufeinander aufbauende Schichten mit jeweils zugehörigen Aufgaben, den einzelnen Schichten zugeordneten Netzwerkprotokollen (vgl. Kapitel 2.1.2.2 und 2.1.3.4) und Schnittstellen zu den übrigen Schichten. Dies sei angeführt als Beispiel für die Anwendung der Referenzmodellierung in einem technischen Zusammenhang ohne direkten Bezug zu Unternehmensorganisation oder Geschäftsprozessen.

5.3.1 Motivation

Die Mehrzahl der bekannten Referenzmodelle hat allerdings entweder einen betriebswirtschaftlichen Bezugsrahmen und zielt auf die Aufbau- und Ablauforganisation von Unternehmen oder dient der Unterstützung von Projekten im Bereich der Softwareentwicklung und zielt auf die Unterstützung entsprechender Prozesse. Sehr wenige Modelle geben *auch* Hinweise zu Architektur und Aufbau von Komponenten (Hard- und Software) im Bereich der IT-Systemtechnik. Wenn ein Referenzmodell solche Aspekte berücksichtigt, dann allerdings *nur am Rande und nicht als primäres Ziel des Modells*. Dies ist das Ergebnis einer umfassenden Untersuchung bereits existierender Modelle, deren Ergebnisse im Folgenden zusammengefasst sind, und damit *die maßgebliche Motivation, ein neues Referenzmodell speziell für die IT-Systemtechnik und den Schwerpunkt Server Based Computing zu entwickeln*.

5.3.2 Untersuchung existierender Referenzmodelle

Die Untersuchung existierender Referenzmodelle, die im Rahmen dieser Arbeit durchgeführt wurde, hatte zum Ziel, Referenzmodelle im Hinblick auf ihren Bezugsrahmen gegenüberzustellen, da dieser direkten Einfluss auf die verwendeten Sprachen und Methoden nimmt. Die Untersuchung greift eine vorhergehende Betrachtung von Referenzmodellen (vgl. [Kett, 2012], S. 39) auf und erweitert diese um für den vorliegenden Kontext relevante Aspekte.

Die Ergebnisse sind in Tabelle 5-1 zusammengefasst. Es wurde dabei unterschieden, ob das jeweilige Referenzmodell die Unternehmensplanung (**U**) im Fokus hat, also einen betriebswirtschaftlichen Ansatz verfolgt und die Aufbau- und Ablauforganisationen von Unternehmen sowie deren Geschäftsprozesse thematisiert, die konkrete Entwicklung (**E**) und Implementierung von Hard- oder Software unterstützen soll und/oder Aspekte der IT-Systemtechnik (**S**) berücksichtigt, also Administratoren bei Konzeption sowie Installation von Client-/ Server-Systemen, Netzwerkkomponenten und Standardsoftware unterstützt.

Die jeweilige Zielrichtung hat direkten Einfluss auf die verwendeten Sprachen. So hatte Holger Kett im Vorfeld seiner Entwicklung eines Referenzmodells zur zielgruppenspezifischen Entwicklung einer webbasierten Informationsplattform für den technischen Vertrieb diverse anderweitige Modelle auf die Eignung für seine Aufgabenstellung untersucht und dabei festgestellt, dass sich im Grunde zwei Teilmengen von Modellen identifizieren lassen, nämlich einerseits solche, die EPK, ERM und Funktionsbäume verwenden, und andererseits solche, die auf die UML setzen (vgl. [Kett, 2012], S. 44). Lediglich ein Modell verwendet zusätzlich die Blockdiagramme der Fundamental Modeling Concepts (FMC) für die Beschreibung der Systemarchitektur. Weitere Erkenntnisse waren, dass nicht alle existierenden Referenzmodelle umfassende methodische Hinweise zur Anwendung der jeweiligen Sprache beinhalten (vgl. [Kett, 2012], S. 41/43) und dass weiterhin der Grad der Evaluierung sehr unterschiedlich ausgeprägt ist.

Im Rahmen dieser Arbeit wurde die Untersuchung aufgegriffen[92], in ihren Bewertungskriterien, wie nachfolgend beschrieben, angepasst und auf weitere Referenzmodelle ausgeweitet. Dabei wurden neben den verwendeten Sprachen auch die Methodik sowie der Grad der Evaluierung und Validierung betrachtet.

Im Hinblick auf die Methodik wird unterschieden, ob das jeweilige Referenzmodell stark ausgeprägte methodische Handreichungen zu seiner Wiederverwendung beinhaltet (●), nur wenige methodische Hinweise bietet (⦿) oder kaum bis keine Methodik umfasst (○).

[92] Drei der ursprünglich von Kett betrachteten Quellen wurden dabei nicht weiter berücksichtigt: So stellt [Kelkar, 2003] einer früheren Erhebung von Fettke und Loos zu Folge kein Referenzmodell im hier verwendeten Sinn dar (vgl. [Fettke, 2004a], S. 7). Das Retail-H Modell (vgl. [Becker, 2004]) ist im Kontext der hier verwendeten Definition eher als Meta-Modell zur Einordnung anderer Modelle zu verstehen und das Supply-Chain-Operations-Reference-Modell (SCOR) zielt speziell auf die Analyse und Beschreibung von Lieferketten und damit verbundenen Geschäftsprozessen innerhalb von und zwischen Unternehmen (vgl. [SCOR, 2010]), was keinen Bezug zu den hier thematisierten IT-Systemen aufweist.

In Bezug auf die Evaluation wurde unterschieden, ob das jeweilige Modell anhand von mehreren (●), lediglich ein bis zwei (◉) oder keinem (○) Anwendungsfall validiert wurde.

Tabelle 5-1: Untersuchung existierender Referenzmodelle

Modell [Quelle]	Beschreibung/Erläuterung	Bezug	Sprachen	Methodik	Evaluierung
Referenzmodellgestütztes Geschäftsprozeßmanagement – Ein Ansatz zur prozeßorientierten Gestaltung vertriebslogistischer Systeme [Kruse, 1996]	Unternehmensplanung, Ablauforganisation, Geschäftsprozessgestaltung und -management im Bereich der Logistik	U	EPK, ERM, Funktionsbäume	●	○
Entwurf eines Referenzmodells für Handelsplattformen im Internet [Frank, 2000]	Ablauforganisation von Geschäftsprozessen, elektronischer Geschäftsverkehr, neue Geschäftsmodelle in Bezug auf Transaktionen über das Internet	U	UML, XML	◉	◉
The Impact of Trading Digital Products on Retail Information Systems [Luxem, 2000]	Unternehmensplanung, Optimierung von Geschäftsprozessen im Bereich des elektronischen Handels von digitalen Produkten.	U	Funktionsbäume	◉	○
Referenzmodellierung für die Handelslogistik [Remmert, 2001]	Unternehmensplanung, Optimierung von Geschäftsprozessen, Logistik im Bereich des stationären Einzelhandels von Lebensmitteln	U	EPK	◉	◉
Referenzmodell zur integrierten Kommunikationsunterstützung von kooperierenden örtlich verteilten Akteuren [Kempf, 2002]	Kommunikationsunterstützung wenig formalisierbarer, ortsunabhängiger synchroner und asynchroner Kooperationen, systematische Unterstützung von System-Analytikern und Designern beim Entwurf kommunikationsunterstützender Systeme	U,E	UML	○	●
Referenzmodell für ein Produktdaten Clearing Center – am Beispiel eines Informationsmodells für die Elektrowirtschaft [Mucha, 2004]	Unternehmensplanung, Optimierung von Prozessen beim zwischenbetrieblichen Austausch qualitätsgesicherter Produktdaten auf Basis eines Produktdaten Clearing Centers	U,E	UML, XML	○	◉
Ein Referenzmodell für die Herstellung von Fachmedienprodukten [Delp, 2006]	Optimierung kaufmännischer Prozesse in inner- und überbetrieblichen Projekten im Bereich der Medienherstellung, Effektivitätsverbesserung von Projektplanung und -management.	U	EPK, ERM	●	●
Referenzmodell zur zielgruppenspezifischen Entwicklung einer webbasierten Informationsplattform für den technischen Vertrieb [Kett, 2012]	Optimierung kaufmännischer Prozesse, Geschäftsmodellentwicklung, Unterstützung von IT-Anbietern und Softwareentwicklern bei der Entwicklung von IT-Lösungen für Handelsvertreter, -vermittler und Hersteller	U,E,S	FMC, EPK, UML und weitere	●	●
Referenzmodell zur Gestaltung der Serviceorganisation in Unternehmen der Raumfahrtbranche zum Betrieb bemannter Raumfahrtsysteme [Forster, 2013]	Unternehmensplanung, Unterstützung von Unternehmen der Raumfahrtbranche bei der Gestaltung und Verbesserung von Serviceprozessen, Optimierung der Ablauforganisation	U	EPK, ERM, Funktionsbäume	●	◉

Quelle: Eigene Darstellung (aufbauend auf [Kett, 2012], S. 43)

Zusammenfassend lässt sich aus der Untersuchung der in Tabelle 5-1 aufgeführten Referenzmodelle ableiten, dass die Mehrheit der bekannten Referenzmodelle einen betriebswirtschaftlichen Schwerpunkt besitzt und auf die Aufbau- und Ablauforganisation von Unternehmen fokussiert. Wenn technische Aspekte hinzukommen, so sind diese dem Bereich der Entwicklung von Softwaresystemen mit fachlichem Bezug zur Betriebswirtschaft zuzuordnen[93]. Bei den verwendeten Sprachen dominieren EPK, ERM und UML. Die Ergebnisse einer früheren Erhebung von über 30 Referenzmodellen stützen diese Erkenntnis und zeichnen ein ebenso heterogenes Bild bezüglich der Tiefe der methodischen Hinweise und des Grades der Validierung anhand von Fallbeispielen oder konkretem Praxiseinsatz der Modelle (vgl. [Fettke, 2004a], S. 17). Dies untermauert die Anforderungen an ein neu zu schaffendes Referenzmodell, zum einen eine Sprache zu wählen, die zur zu beschreibenden Domäne passt, und zum anderen umfassende Hinweise zu geben, wie diese zu verwenden ist.

5.3.3 Auswahl der Sprache

Wird als gegebenes Ziel vorausgesetzt, den *Aufbau* einer Server Based Computing Infrastruktur – also die Implementierung und Verschaltung von Hard- und Softwarekomponenten im Bereich der IT-Systemtechnik – zu beschreiben, so ist zu konstatieren, dass die im Rahmen vieler anderweitiger Referenzmodelle verbreiteten Sprachen nicht geeignet sind, dieses Ziel zu erreichen, da dort i. d. R. nicht *Aufbau* sondern *Abläufe* und *Datenstrukturen* im Vordergrund stehen.

Die EPK dienen primär einer semiformalen Darstellung von Geschäftsprozessen bzw. Abläufen in und zwischen Unternehmen. Sie sind daher im Bereich der Geschäftsprozessmodellierung zu verorten und helfen nicht, das zuvor genannte Ziel zu erreichen. ER Diagramme dienen der Modellierung von Daten und ihren Beziehungen in einem bestimmten Kontext, der betriebswirtschaftlicher oder technischer Natur sein kann, sind damit der Software- bzw. Datenbankentwicklung zuzuordnen und somit im vorliegenden Kontext ebenfalls nicht zielführend. Auch die UML ist dem Bereich der Softwareentwicklung zuzurechnen und dient der grafischen Darstellung von Datenstrukturen und sequenziellen Abläufen, die im Kontext von Entwicklungsprojekten zu implementieren sind.

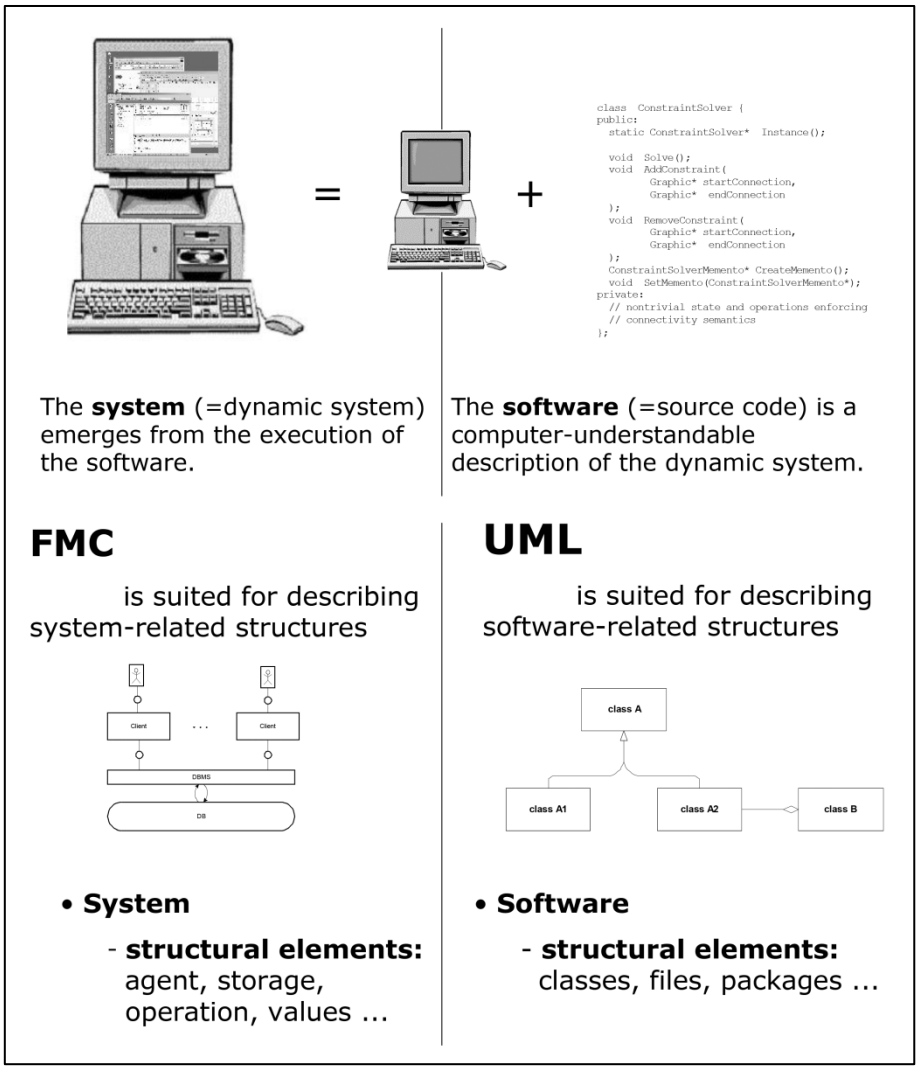
Damit verbleiben aus der Erhebung existierender Referenzmodelle als nützlichster Ansatz die Blockdiagramme der **Fundamental Modeling Concepts (FMC)** (vgl. [Keller, 2003]), welche im Folgenden zur Beschreibung von Server Based Computing Infrastrukturen eingesetzt werden sollen.

Diese Auswahl begründet sich darin, dass die FMC explizit nicht als Konkurrenz oder Alternative zur UML, sondern als deren Komplement verstanden werden dürfen (vgl.

[93] Auch das eingangs – als Beispiel für ein Referenzmodell mit rein technischem Hintergrund – angeführte ISO/OSI-7-Schichten-Modell (vgl. [ISO, 1994]) zielt auf die Entwicklung von u. a. Software, Protokollen, Netzwerk-Technologien. Soweit formale Sprachkonstrukte zum Einsatz kommen (vgl. [Facchi, 1995], S. 7), sind diese speziell auf diesen Zweck ausgerichtet und für den hier vorliegenden Fall der Beschreibung von Server Based Computing Infrastrukturen nicht geeignet.

[Apfelbacher, 2004]). Die UML dient der Beschreibung der Software-Strukturen eines Modellierungsgegenstandes, während die FMC auf die Systemstrukturen fokussieren (Abbildung 5-6). Beide Sprachen ergänzen einander somit. Die UML ist sehr gut geeignet, um insbesondere im Rahmen objektorientierter Softwareentwicklung Abläufe und Zusammenhänge zu beschreiben, wird in der Praxis jedoch nicht angewendet, um Aufbau und Architektur von IT-Systemen und -infrastrukturen darzustellen: „UML is not and can not be used to describe systems. Nevertheless concepts for this are needed." (vgl. [Apfelbacher, 2004], S. 17).

Abbildung 5-6: FMC und UML



Quelle: [Apfelbacher, 2004]

Diese Lücke schließen die FMC, indem sie den direkten Bezug zu Architektur und Aufbau eines Systems herstellen und so „(…) die bestmögliche Unterstützung der Kommunikation zwischen Menschen über jegliche Art informationeller Systeme mittels didaktischer Modelle (…)" (vgl. [Knöpfel, 2004], S. 25) bieten.

Wie Knöpfel weiter ausführt, differenzieren die FMC drei verschiedene Sichten eines Systems, die jeweils eigene sprachliche Konstrukte zu ihrer Beschreibung verwenden (vgl. [Knöpfel, 2004], S. 25 und auch [Rhenau, 2004]):

- **Kompositionelle Struktur:** Diese Sicht beschreibt Aufbau und Architektur eines Systems in Form von Blockdiagrammen, wobei Akteure durch Rechtecke, Speicher in Form von Kreisen, Kommunikationsbeziehungen zwischen Akteuren und Speicherzugriffe durch Pfeile symbolisiert werden[94] (siehe auch Kapitel 10).

- **Dynamische Struktur:** Diese Sicht beschreibt sequenzielle Abläufe und nutzt erweiterte Petri-Netze zur Darstellung über der Zeit veränderlicher Zustände eines Systems.

- **Wertebereichsstruktur:** Diese Sicht stellt die Struktur der in einem System gespeicherten und verarbeiteten Informationen dar und nutzt dazu ER Diagramme.

Während letztere Perspektive eher der Softwareentwicklung zuzurechnen ist, sind die ersteren beiden und darunter *insbesondere die kompositionelle Struktur mit ihren Blockdiagrammen* geeignet, auch im Bereich der IT-Systemtechnik zu unterstützen.

Neben der Anwendung im Kontext des Referenzmodells zur zielgruppenspezifischen Entwicklung einer webbasierten Informationsplattform für den technischen Vertrieb (vgl. [Kett, 2012], S. 55, 67, 76) sind weitere Beispiele für den Einsatz der FMC bekannt. So kamen die FMC auch bei Design und Implementierung des Apache Web Servers zum Einsatz (vgl. [Gröne, 2008]) sowie im Rahmen der Implementierung von virtuellen Maschinen auf Basis von Java-Technologien (vgl. [Kugel, 2005]). Auch im Kontext des SAP Referenzmodells finden entsprechende sprachliche Konstrukte Verwendung (vgl. [SAP, 2007]).

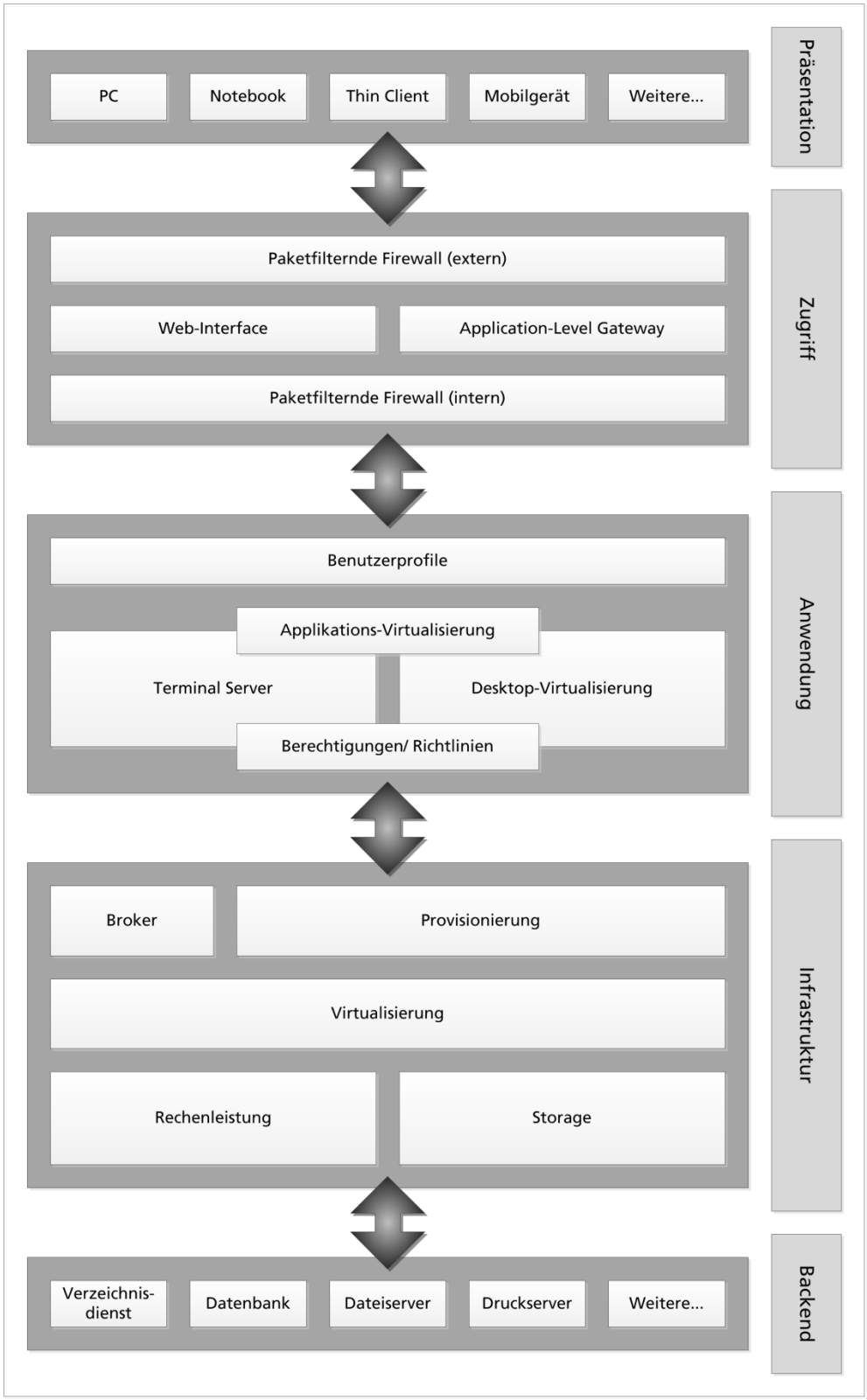
Im folgenden Kapitel wird nun die Entwicklung des Referenzmodells für SBC Infrastrukturen beschrieben, wobei die FMC Blockdiagramme zum Einsatz kommen und methodische Hinweise zu deren Verwendung erläutert werden.

5.4 Referenzmodell für SBC Infrastrukturen

Basierend auf den vorhergehenden Ausführungen zu Architektur und technischen Details der Implementierung von SBC Infrastrukturen in den Kapiteln 2 und 3 (vgl. insbesondere Abbildung 3-11 in Kapitel 3.3) wird als *zentrales Ergebnis der Arbeit* das folgende Referenzmodell als hersteller- und produktneutrale *Gesamtsicht* auf eine SBC Infrastruktur aufgebaut (Abbildung 5-7):

[94] Die komplette Syntax ist dokumentiert unter http://www.f-m-c.org/notation_reference (abgerufen am 01.02.2016). Weitere Empfehlungen zur Verwendung der Sprachkonstrukte finden sich in den „FMC Visualization Guidelines" (vgl. [Apfelbacher, 2005]), die auch unter http://www.fmc-modeling.org/visualization_guidelines (abgerufen am 01.02.2016) verfügbar sind.

Abbildung 5-7: Referenzmodell für SBC Infrastrukturen (Gesamtsicht)



Quelle: Eigene Darstellung

Die Konstruktion des Referenzmodells folgt einem „Top-Down"-Ansatz und wird beginnend beim Kunden und seinen Anforderungen über den Zugriffsweg, die eigentlichen Anwendungen bis hin zur nötigen Infrastruktur sowie den erforderlichen Komponenten und Diensten im Backend aufgebaut. Der folgende Abschnitt 5.4.1 beschreibt die fünf Schichten des Referenzmodells im Detail.

Wenngleich in den vorherigen Kapiteln vielfach der Schwerpunkt auf Lösungen und Produkten der Hersteller Citrix und Microsoft lag, erfüllt dieses Modell einen allgemeingültigeren Anspruch. Die Allgemeingültigkeit lässt sich sogar soweit fassen, dass das Referenzmodell nicht auf die Welt der Windows Betriebssysteme beschränkt ist, sondern auch als Beschreibung für vergleichbare Architekturen auf Basis von Linux bzw. anderen Unix-artigen Betriebssystemen verwendbar ist.

5.4.1 Die fünf Schichten des Referenzmodells

Das Referenzmodell untergliedert eine SBC Infrastruktur in fünf Schichten[95]. Diese sind dem „Top-Down"-Ansatz folgend von oben nach unten betrachtet:

5. **Präsentation:** Auf dieser Schicht erfolgen die Darstellung der von den Endanwendern gewünschten Anwendungen und die Interaktion mit diesen nach dem Prinzip der entfernten Präsentation. Hier sind entsprechend die verschiedenen Ausprägungen von Client-Hardware und -Software sowie ggf. weitere Client-seitige Peripherie-Geräte angesiedelt, welche über ein Remote-Protokoll mit den Ressourcen auf der Server-Seite kommunizieren.

4. **Zugriff:** Diese Schicht vermittelt die Netzwerkverbindungen zu den Client-Computern der Endanwender. Die Absicherung der Kommunikation erfolgt dabei in der Regel nach dem Prinzip „P-A-P" (Paketfilter – Application-Level Gateway – Paketfilter, vgl. Kapitel 3.2.4). Ein Web-Interface in Form einer interaktiv aufzurufenden Webseite oder eines Webservices übernimmt die Authentifizierung der Anwender sowie die Enumeration der für sie freigegebenen Ressourcen aus der unterliegenden Anwendungsschicht.

3. **Anwendung:** Dieser Schicht werden die Server, Techniken und Dienste zugeordnet, welche direkt der Ausführung der von den Endanwendern gewünschten Applikationen dienen. Dies umfasst auch die VM der Terminal Server und virtuellen Desktops selbst. Unterliegend kommen Konzepte zu deren Absicherung, wie Berechtigungen und zentral definierte Richtlinien, Skripte usw., hinzu (vgl. Kapitel 3.3.1). Darüber liegen ggf. Techniken zur Virtualisierung von Applikationen und zur Verwaltung von Benutzerprofilen sowie individuellen Einstellungen.

[95] Aus Gründen der Übersichtlichkeit wird darauf verzichtet, granular sämtliche Abhängigkeiten und Kommunikationsbeziehungen zwischen den Servern und weiteren Komponenten der einzelnen Schichten untereinander und mit den anderen Schichten in der Grafik darzustellen.

2. **Infrastruktur:** Diese Schicht beinhaltet sämtliche Hard- und Software, welche die Basis für den Betrieb von Terminal Servern oder virtuellen Desktops bildet. Hier sind unter dem Punkt der Rechenleistung die Verarbeitungskapazitäten in Form von physischen Servern mit Prozessoren, Hauptspeicher usw. angesiedelt. Dabei kann es sich um dedizierte Rack-Server im 19-Zoll-Format oder um integrierte Lösungen, wie die in Kapitel 2.4.2 diskutierten Blade-Systeme, handeln. In der Regel verfügen diese Server-Systeme nur über Festspeicher geringer Kapazität zur Aufnahme eines Hypervisors und binden zusätzlichen Storage per iSCSI, NFS oder Fiber Channel an. In diesem Storage werden die virtuellen Festplatten der VM, Images, Snapshots usw. gespeichert. Der Hypervisor wird durch die darüber liegende „Virtualisierung" dargestellt. Ebenfalls dieser Schicht zugeordnet werden Server, die als Broker bzw. Controller zur Verwaltung der Terminal Server und virtuellen Desktops dienen, sowie sämtliche Komponenten und Techniken, die zur Provisionierung der zugehörigen VM benötigt werden.

1. **Backend:** In dieser Schicht befinden sich sämtliche Hintergrund- und Hilfsdienste, welche die SBC Infrastruktur für ihr korrektes Funktionieren benötigt. Dazu zählt ein Verzeichnisdienst für die Verwaltung von Benutzer- und Computer-Konten sowie deren Authentifizierung. In der Regel wird es sich dabei um ein Microsoft Active Directory (AD) oder ggf. einen anderweitigen Lightweight Directory Access Protocol (LDAP) kompatiblen Dienst handeln. Zusätzlich vorgesehen ist hier mindestens ein Datenbankserver, der die Konfiguration von Diensten der anderen Schichten speichert, soweit diese eine Datenbank benötigen. Im Backend ist weiterhin ein Dateiserver aufgeführt, der an dieser Stelle zur Speicherung der Nutzdaten[96] und ggf. Benutzerprofile der Endanwender vorgesehen ist. Hinzu kommen weitere Dienste, wie etwa Druckserver, E-Mail- oder Web-Server, welche von den Applikationen auf der Anwendungs-Schicht benötigt werden.

Das Referenzmodell bildet somit sämtliche Elemente einer typischen SBC Infrastruktur ab, die vorhanden sein können, wenn mit (Server-)Virtualisierung, den klassischen Terminal Servern, der neueren Desktop-Virtualisierung, ergänzenden Technologien, wie z. B. der Applikations-Virtualisierung, und sicherem Zugriff von außen alle verfügbaren Techniken und Werkzeuge zum Einsatz kommen.

5.4.2 Ableitung konkreter Informationsmodelle

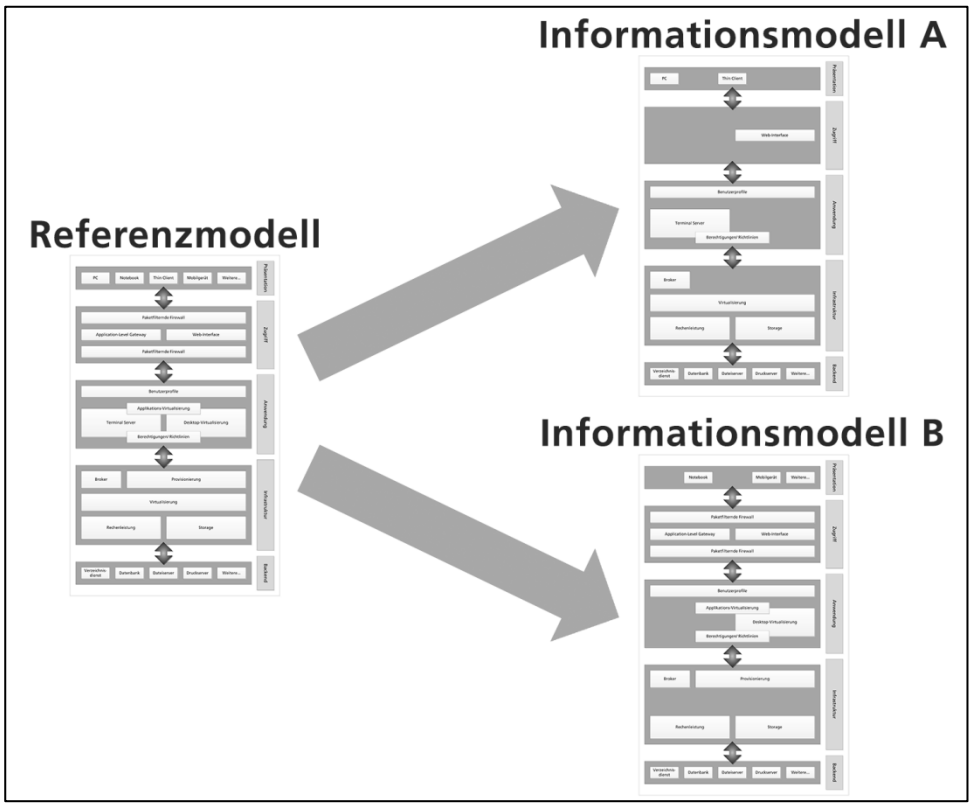
Bei der Ableitung eines Informationsmodells für eine konkrete SBC Infrastruktur *können* alle Module des Referenzmodells verwendet werden, *müssen* dies jedoch nicht zwingend. Das Referenzmodell stellt damit einen modularen Baukasten zur Verfügung, aus dem sich Systemtechniker bei der Konzeption einer neuen SBC Infrastruktur oder der Weiterentwicklung einer bestehenden je nach Bedarf bedienen können.

[96] Der Begriff der „Nutzdaten" wird hier in dem Sinn verwendet, dass es sich um die Daten handelt, die vom Anwender unmittelbar für seine Arbeitsinhalte benötigt und genutzt werden, also etwa ein Microsoft Word-Dokument oder eine Excel-Tabelle. In Abgrenzung dazu ist zur Speicherung sämtlicher Verwaltungsdaten, virtueller Festplatten, Images usw. mit dem „Storage" ein separater Speicherbereich in der Infrastruktur-Schicht ausgewiesen.

So ist es etwa denkbar, dass ein Unternehmen ausschließlich Anwendungen auf Basis von Terminal Servern bereitstellen möchte und das auch nur an stationäre PC und Thin Clients, die sich direkt am Standort bzw. im inneren Netzwerk des Unternehmens befinden. Da das Unternehmen nur sehr wenige gleichzeitige Benutzer zu bedienen hat, sollen die Terminal Server zudem unter Verzicht auf Techniken der Provisionierung direkt als manuell verwaltete VM installiert werden („Informationsmodell A" in Abbildung 5-8). In diesem Fall können mehrere Module des Referenzmodells entfallen, darunter die Provisionierung, Desktop- und Anwendungsvirtualisierung sowie die Techniken zur Absicherung des Zugriffs von außen und die nicht zutreffenden Typen von Clients.

In einem anderen Szenario könnte die Anforderung dagegen lauten, Notebooks und anderweitige mobile Clients, die sich oft außerhalb des sicheren internen Netzes eines Unternehmens befinden, mit Anwendungen zu versorgen. Da es sich um zu Terminal Servern inkompatible Anwendung handelt, soll die Bereitstellung auf Basis virtueller Desktops erfolgen. Und da die Anwendungen, weiterhin angenommen, besonders hohe Anforderungen bezüglich der Rechenleistung stellen, sollen die Desktops ohne Virtualisierung direkt auf Hardware provisioniert werden („Informationsmodell B" in Abbildung 5-8). Auch in diesem Fall würden mehrere Module des Referenzmodells nicht benötigt, nämlich die Virtualisierung, die Terminal Server sowie die Client-Typen der PC und Thin Clients.

Abbildung 5-8: Referenz- und Informationsmodelle für SBC Infrastrukturen



Quelle: Eigene Darstellung

Diese zwei Anwendungsfälle seien als Beispiele dafür angeführt, dass aus der Vorlage des Referenzmodells nahezu beliebige Umgebungen konstruiert werden können und die Allgemeingültigkeit für möglichst viele praktische Anwendungsfälle somit gegeben ist. Alle Anwendungsfälle, welche sich durch die Architektur des Referenzmodells beschreiben lassen, bilden die *Klasse der SBC Infrastrukturen*.

5.4.3 Methodik und Verwendung der FMC Sprachelemente

Um nun aus dem abstrakten Referenzmodell konkretere Informationsmodelle zur Beschreibung von SBC Infrastrukturen zu gewinnen, wird ein „Top-Down"-Ansatz gewählt, d. h. die Schichten des Referenzmodells werden von oben nach unten betrachtet und anhand von Leitfragen genauer spezifiziert. Die Gegebenheiten der jeweiligen Schicht werden dabei mittels der FMC Blockdiagramme notiert, um so zu einer genauen Beschreibung der gesamten Infrastruktur zu gelangen. Die grundlegenden Sprachelemente der FMC Blockdiagramme und Hinweise zu ihrer Verwendung sind in Appendix A beschrieben. Im Folgenden wird ihre Anwendung im Hinblick auf SBC Infrastrukturen exemplarisch für Teilaspekte eines Informationsmodells erläutert. Die umfassende Abbildung eines Beispielszenarios folgt im Rahmen des Einsatzes und der Validierung in Kapitel 6.

Der „Top-Down"-Ansatz der Methodik beginnt auf der fünften Schicht des Referenzmodells, also direkt bei den Kunden bzw. Endbenutzern der SBC Infrastruktur, und hilft somit, ein neu zu implementierendes oder weiterzuentwickelndes System auf allen Schichten optimal an deren Anforderungen auszurichten. Dabei sind die folgenden Leitfragen und Aspekte zu berücksichtigen:

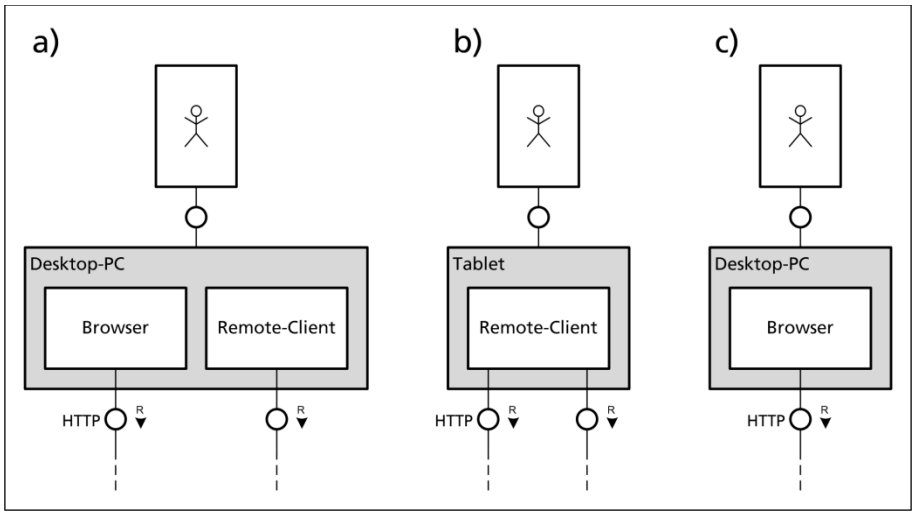
5.4.3.1 Präsentationsschicht

Die Analyse der Anforderungen beginnt mit den Endgeräten der Kunden. *Welche Client-Computer* setzen die Kunden ein? Handelt es sich um klassische Arbeitsplatz-Computer, also Desktop-PC oder Notebooks mit den Betriebssystemen Microsoft Windows, Apple Mac OS X oder einer Linux-Distribution? Sollen mobile Clients, also Tablets oder Smartphones mit den Betriebssystemen Apple iOS, Google Android oder Microsoft Windows Phone/Mobile/RT zum Einsatz kommen? Sollen Thin Client Computer (vgl. Kapitel 2.4.1) mit einem für den Zugriff auf SBC Infrastrukturen optimiertem Betriebssystem, wie Microsoft Windows ThinPC/ Embedded oder einem angepassten Linux, Verwendung finden?

Unabhängig vom jeweiligen Endgerät ist die Frage zu klären, ob es gewünscht und möglich ist, ein dediziertes Client-Programm zu installieren. Dies sollte i. d. R. der Fall sein, da nur so eine optimale Unterstützung der auf den tieferen Schichten gebotenen Funktionalität der SBC Infrastruktur zu realisieren ist. Falls es auf Grund von technischen Beschränkungen oder organisatorischen Fragen nicht möglich sein sollte, ein Client-Programm zu installieren, müsste der Zugriff auf die tieferen Schichten als rein Browser-basierte Lösung umgesetzt werden. Diese Fragen berühren also auch Produktauswahl und technische Entscheidungen bezüglich der Implementierung auf den tieferen Schichten, da die SBC Infrastruktur die Anforderungen der Client-Seite erfüllen muss.

Die Abbildung 5-9 zeigt verschiedene Möglichkeiten, die Benutzer und ihre Endgeräte in FMC Blockdiagrammen zu notieren: Die Benutzer interagieren jeweils mit ihren Client-Computern. Dabei kann es sich beispielsweise (**a**) um einen Desktop-PC handeln, auf dem der Benutzer zunächst in einem Browser per HTTP eine interaktive Webseite aufruft, welche die für ihn veröffentlichten Anwendungen auflistet. Nachdem er eine Anwendung ausgewählt und gestartet hat, vermittelt das Web-Interface eine Sitzung zu einem Terminal Server oder virtuellen Desktop, auf dem ein Remote-Client der auf den tieferen Schichten eingesetzten Lösung dann eine Sitzung über das entsprechende Protokoll öffnet.

Abbildung 5-9: FMC Notation verschiedener Clients



Quelle: Eigene Darstellung

Alternativ könnte (**b**) der Remote-Client – z. B. eine App auf einem Tablet oder Smartphone – auch direkt per HTTP mit einem Webdienst in der Zugriffsschicht kommunizieren und die Liste der veröffentlichten Anwendungen anfordern, um diese dann zu starten. Weiterhin ist auch der Client-lose Zugriff (**c**) abbildbar. In diesem Fall müsste dann das Web-Interface das jeweilige Remote-Protokoll in HTML5 kapseln[97].

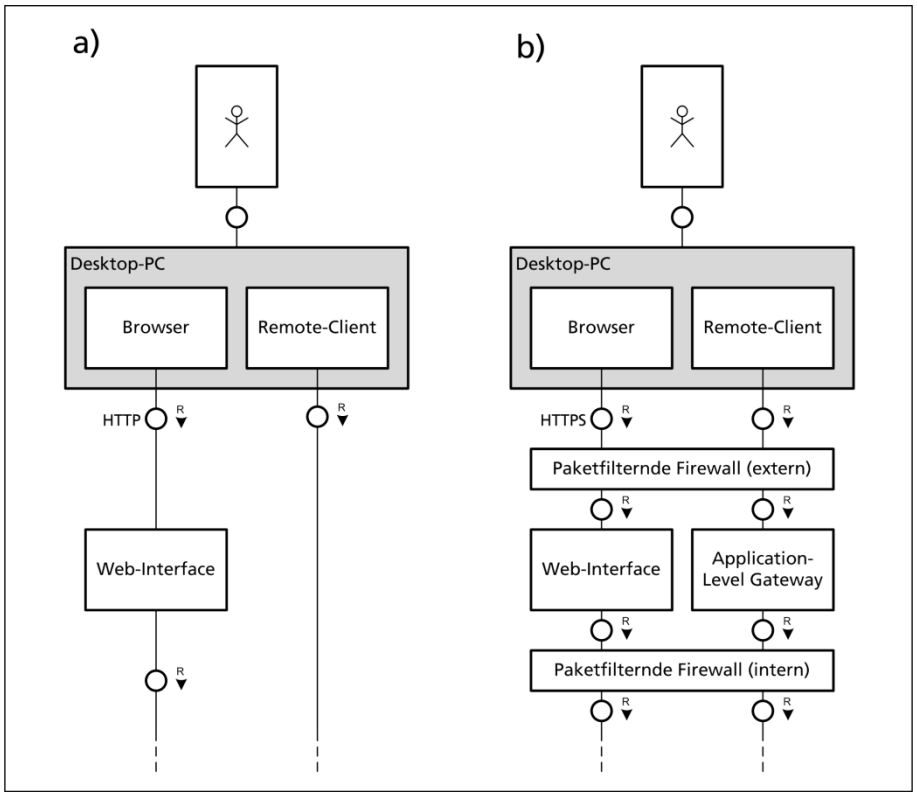
Weitere zu beantwortende Fragen leiten über zur nächstgelegenen Schicht des Modells, der Zugriffsschicht: Sind die Clients stationär oder an wechselnden Standorten im Einsatz? Wo befinden sich die Clients geografisch bzw. logisch in Bezug auf die Netzwerk-Topologie?

[97] Im Falle von Implementierungen auf Basis von Citrix-Technologien wird dies z. B. über den Citrix Receiver for HTML5 realisiert, der das Protokoll ICA in einen H.264-kodierten Videostrom kapselt, so dass es von einem HTML5-fähigen Browser ohne weitere Installation eines nativen Client-Programms verarbeitet werden kann: http://docs.citrix.com/en-us/receiver/html5/1-7/receiver-html5-17-about.html (abgerufen am 01.02.2016).

5.4.3.2 Zugriffsschicht

Die zuvor genannten Fragen sind relevant für Design und Implementierung der Netzwerk-Komponenten auf der Zugriffsschicht. *Von wo* greifen die Kunden auf die Infrastruktur zu? Befinden sich die Endgeräte gemeinsam mit der übrigen SBC Infrastruktur am gleichen Standort? Sind die Endgeräte im gleichen Netzsegment oder im Sinne der Informationssicherheit in gleichrangigen Netzsegmenten, zwischen denen ein direkter[98] Informationsaustausch möglich ist? Sind für die Clients transparente VPN-Tunnel zwischen verschiedenen Netzsegmenten bzw. Standorten zu berücksichtigen? Oder erfolgt der Zugriff auf die SBC Infrastruktur aus Client-Sicht gar über komplett unsichere Netze, sprich das öffentliche Internet, so dass zusätzliche Sicherungsmaßnahmen durch den Einsatz von paketfilternden Firewalls, Application-Level Gateways und VPN-Technologien (vgl. Kapitel 3.2) zu etablieren sind?

Abbildung 5-10: FMC Notation der Zugriffsschicht



Quelle: Eigene Darstellung

Im einfachsten Fall (Abbildung 5-10, (**a**)) ist den Clients ein transparenter Zugriff auf die Systeme der tieferen Schichten möglich. Dann ist im Kontext der Zugriffsschicht lediglich das

[98] „Direkter" Zugriff meint hier eine auf der Vermittlungsschicht des ISO/OSI-Referenzmodells geroutete Kommunikation, ohne dass Firewalls oder anderweitige Technologien zu berücksichtigen wären.

Web-Interface zu berücksichtigen, welches die verfügbaren Applikationen enumeriert und an den Client-Computer liefert. Dieser nimmt direkten Kontakt mit den Terminal Servern und virtuellen Desktops in der Anwendungsschicht auf.

Bei einem Zugriff über unsichere Netze (**b**) ist die Zugriffsschicht entsprechend abzusichern. Mindestens dann sollte HTTPS für Browsersitzungen und eine Kapselung in SSL für das Remoteprotokoll zum Einsatz kommen. Die zusätzlich nötigen Komponenten werden im FMC Blockdiagramm als Akteure notiert, die untereinander mittels Kanälen kommunizieren.

Je nach Bedarf können in gleicher Weise weitere Netzwerkkomponenten tieferer ISO/OSI-Schichten, wie Router und Switches, notiert werden, sofern sie im jeweiligen Anwendungsfall relevant sind.

5.4.3.3 Anwendungsschicht

Auf der Anwendungsschicht steht die Frage im Vordergrund, *was* die Kunden innerhalb der SBC Infrastruktur tun wollen. *Welche Applikationen* führen die Endanwender aus? Die Applikationen geben zunächst grundsätzlich die auf der Anwendungsschicht einzusetzenden Betriebssysteme vor. Handelt es sich um Applikationen, die eine bestimmte Version von Microsoft Windows oder eine Linux-Distribution voraussetzen?

Weiterhin ist zu entscheiden, ob Terminal Server oder virtuelle Desktops zum Einsatz kommen sollen. Sind die Applikationen zur Ausführung auf Multi-User-Betriebssystem, d. h. Terminal Servern, geeignet? Oder setzen Sie auf Grund von Lizenzbedingungen oder Anforderungen an die Performance, Kompatibilität, individuelle Anpassbarkeit oder Sicherheit ein dediziertes Client-Betriebssystem pro Benutzer voraus?

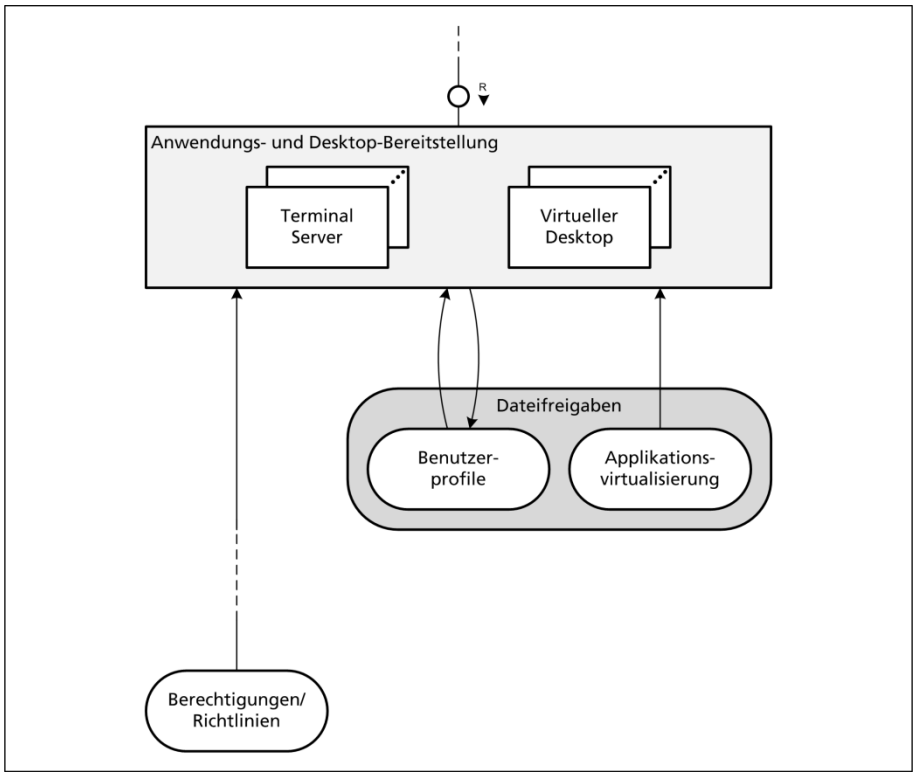
Darüber hinaus ist Auswahl und Einsatz der begleitenden Technologien und Komponenten – Benutzerprofile, Applikations-Virtualisierung sowie Berechtigungen und Richtlinien – zu klären. Treiber sind hierfür zunächst wiederum die Anforderungen seitens der Applikationen: Gibt es Bedarf für Individualisierung der Arbeitsumgebung und inwieweit können entsprechende Einstellungen über die Benutzerprofile oder Richtlinien sinnvoll vorbelegt und gespeichert werden? Reichen die im jeweiligen Betriebssystem vorhandenen Möglichkeiten des Benutzerprofil-Managements oder gibt es Bedarf für den Einsatz zusätzlicher Lösungen von Drittanbietern, etwa weil die im Betriebssystem vorhandenen Methoden bestimmte Einstellungen nicht speichern können?

Können oder sollen zur Bereitstellung der Anwendungen zusätzlich Techniken der Applikations-Virtualisierung (vgl. Kapitel 2.3.3) zum Einsatz kommen? Anlass hierfür kann z. B. sein, dass eine bestimmte Applikation nur selten benutzt wird oder häufig aktualisiert werden muss und hierzu nicht laufend die Images von Terminal Servern und virtuellen Desktops angepasst werden sollen. Ein weiterer Anhaltspunkt können Aspekte der Kompatibilität oder Sicherheit sein, die den Einsatz von Techniken der Applikations-Virtualisierung nahelegen.

Überlegungen zum Einsatz von Berechtigungen und Richtlinien ergeben sich neben den Anforderungen der jeweiligen Applikationen zudem aus technischen und organisatorischen Randbedingungen, die den Anwendern den Umgang mit dem System erleichtern oder aber bestimmte Möglichkeiten nach Vorgaben des Unternehmens einschränken sollen. Beispiele hierfür sind die Vorgabe eines Proxy-Servers, so dass Benutzer diesen nicht selbst eingeben müssen, oder in Microsoft Windows Umgebungen die Einschränkung des Zugriffs auf System-Partition und -Registrierung mittels Gruppenrichtlinien.

Eine mögliche Notation als FMC Blockdiagramm zeigt Abbildung 5-11. In diesem Beispiel sind mehrere Terminal Server und virtuelle Desktops logisch zu einer Anwendungs- und Desktop-Bereitstellung gruppiert. Auf diese Gruppe greifen die Clients über die höheren Schichten mittels Remote-Protokoll zu. Die Terminal Server und Desktops lesen Benutzerprofile aus einer Dateifreigabe und schreiben Änderungen dorthin zurück. Abbilder virtualisierter Applikationen werden aus einer anderen Freigabe nur gelesen. Vorgaben für Berechtigungen und Richtlinien werden aus einem anderweitigen Speicher gelesen, wobei es sich hier nicht zwingend um eine Dateifreigabe handeln muss. In Microsoft Windows Umgebungen dienen typischerweise die Active Directory Domain Controller in der tieferen Backend-Schicht als Speicherort für Gruppenrichtlinien, welche Vorgaben zu Berechtigungen und anderen Systemeinstellungen von zentraler Stelle verteilen.

Abbildung 5-11: FMC Notation der Anwendungsschicht



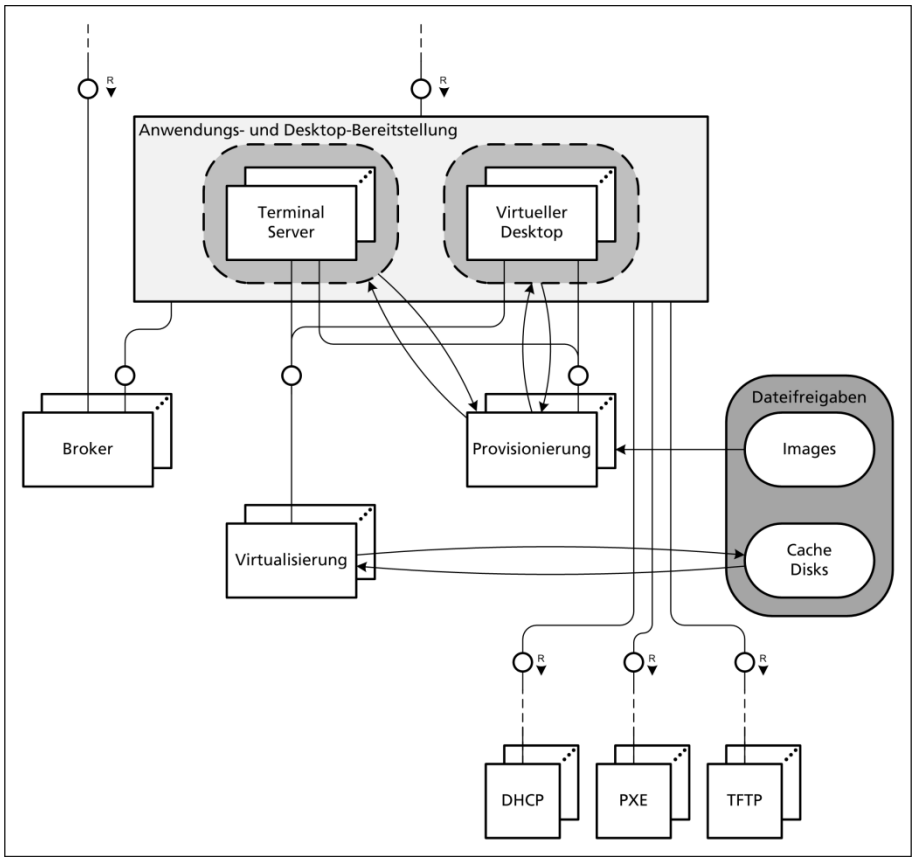
Quelle: Eigene Darstellung

Weitere Fragen bezüglich Kapazitäts- und Leistungsanforderungen der Applikationen leiten über zur Infrastruktur-Schicht.

5.4.3.4 Infrastruktur-Schicht

Hier steht die Frage im Vordergrund, *wie viele Benutzer* auf die SBC Infrastruktur zugreifen werden, insbesondere *wie viele Benutzersitzungen und Applikationen* in der Spitze gleichzeitig zu bedienen sind. Wie viele (virtuelle) Prozessoren, wie viel Arbeits- und Festplattenspeicher benötigt eine Applikation? Hieraus lässt sich ableiten, wie viele Terminal Server oder virtuelle Desktops mit welcher Ausstattung bereitzustellen sind. Sind nur sehr wenige Benutzer zu bedienen, eignen sich u. U. einige wenige dediziert installierte Terminal Server oder virtuelle Desktops. In der Regel wird sich jedoch – allein aus Gründen der einheitlichen Bereitstellung und des Update-Managements – der Einsatz von Provisionierungstechnologien (vgl. Kapitel 2.3.1) empfehlen.

Abbildung 5-12: FMC Notation der Infrastruktur-Schicht



Quelle: Eigene Darstellung

Eine exemplarische Notation einer Provisionierungsumgebung als FMC Blockdiagramm – orientiert an der Funktionsweise der Citrix Provisioning Services (PVS) – ist in Abbildung 5-12 dargestellt. In diesem Fall ist die Anzahl der Terminal Server und virtuellen Desktops dynamisch. Beide Typen von Systemen werden von einem oder mehrere Provisionierungsservern verwaltet,

welche die Basisimages aus einer Dateifreigabe laden und den Systemen zur Laufzeit zur Verfügung stellen. Dazu benötigen die Systeme allerdings mit DHCP-, ggf. PXE- und TFTP-Server weitere Hilfsdienste aus dem Backend.

Die Systeme werden auf einem oder mehreren Virtualisierungshosts betrieben, welche den Systemen zusätzlich virtuelle Festplatten als Cache bereitstellen, um Änderungen gegenüber dem Basisimage speichern zu können. Die Broker oder Controller speichern Informationen darüber, welche Desktops oder Applikationen für die Endanwender freigegeben sind, liefern diese Informationen an das Web-Interface in der Zugriffsschicht und vermitteln die Endanwender an die einzelnen Zielsysteme.

Ggf. sind weitere Rahmenbedingungen zu berücksichtigen, die sich durch die auszuführenden Applikationen ergeben. Sind spezielle Anforderungen seitens der Applikationen zu erfüllen, z. B. im Hinblick auf 3D-Grafikbeschleunigung oder den Zugriff auf anderweitige Hardware, wie USB-Dongle oder besondere Steckkarten?

Die zuvor genannten Fragen erleichtern Konzeption und Implementierung der unterliegenden Computing-, Storage- und Netzwerkkomponenten. Eignen sich die Applikationen für die Virtualisierung oder sollen die Terminal Server und virtuellen Desktops direkt auf physische Zielsysteme installiert bzw. provisioniert werden? Wie viele (Virtualisierungs-)Hosts mit welcher Hardwareausstattung sollen zum Einsatz kommen? Und welche Art von Storage und Netzwerkanbindung benötigen diese Systeme?[99] Um das Bild zu vervollständigen, ist im letzten Schritt nun noch die Backend-Schicht zu integrieren.

5.4.3.5 Backend-Schicht

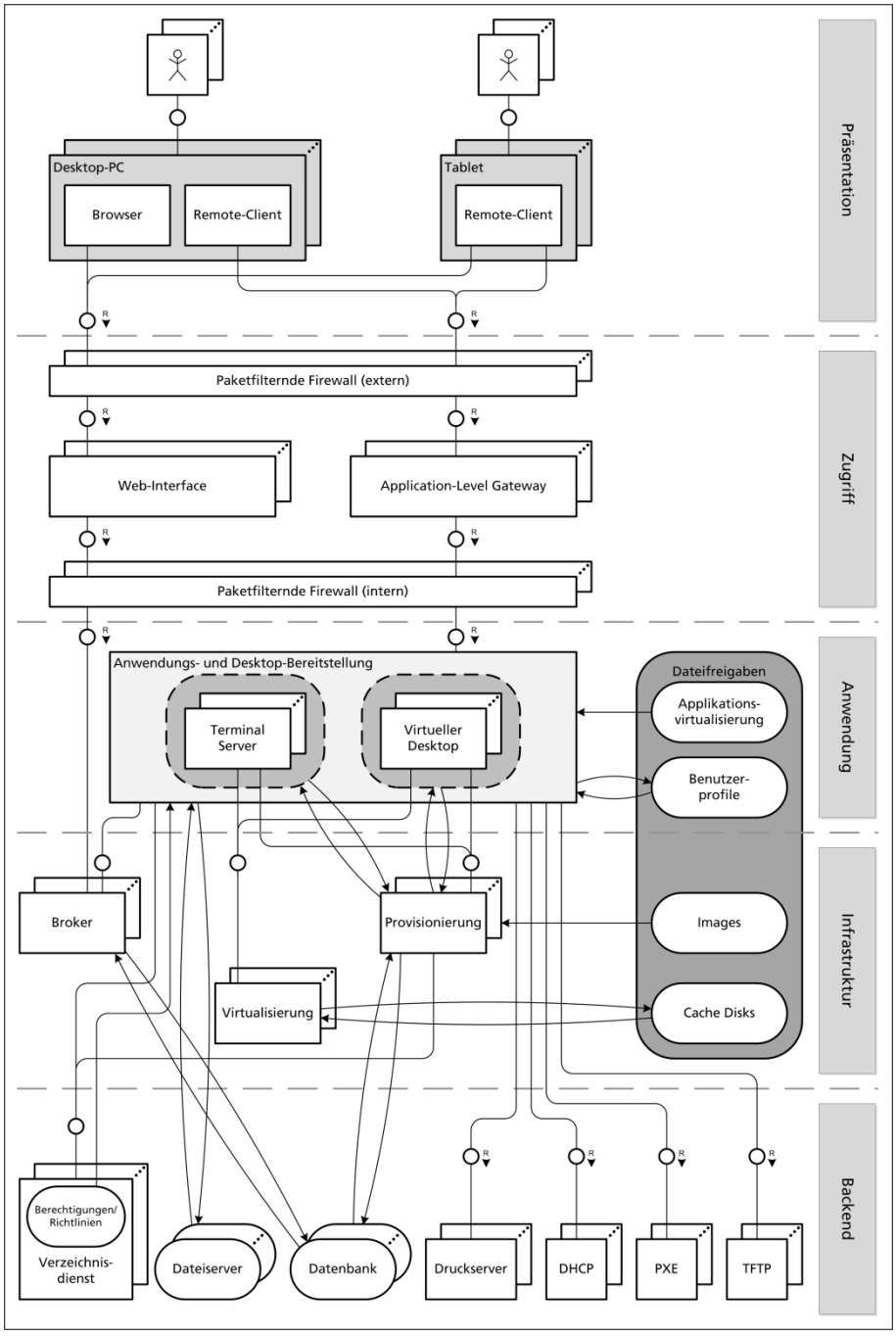
Die in der Backend-Schicht zu implementierenden Systeme und Dienste ergeben sich einerseits aus den für die Infrastruktur- und Anwendungsschichten ausgewählten Lösungen: Benötigen diese einen Verzeichnisdienst, speziell ein Microsoft Active Directory? Benötigen die etwaigen Technologien zur Provisionierung, die Terminal Server und virtuellen Desktops bzw. die Applikationen selbst Datenbanken? Sind, wie im vorhergehenden Kapitel bereits angeführt, DHCP-, PXE- und TFTP-Server erforderlich? Andererseits sind ggf. weitere von den Applikationen bzw. Endanwendern benötigte Dienste zu berücksichtigen: Benötigen die Applikationen bzw. die Endanwender direkten Zugriff auf Dateiserver zur Ablage von Nutzdaten? Welche weiteren Server-Dienste, wie z. B. E-Mail-, Web- oder Lizenzserver sind erforderlich?

Die Abbildung 5-13 zeigt die Gesamtsicht auf *ein mögliches* konkretes Informationsmodell, das sich mittels der Sprachelemente der FMC Blockdiagramme beschreiben lässt. In diesem Fall

[99] Insbesondere diese Fragen können im ersten Durchlauf des Lebenszyklus (vgl. Kapitel 4.3) natürlich nur an Hand von begründeten Annahmen oder durch Erfahrungswerte angenähert werden. Ihre Prüfung ist Gegenstand der weiteren Verwendung des Informationsmodells (vgl. Kapitel 5.4.4), woraufhin weitere Durchläufe des Lebenszyklus das Modell und seine Implementierung konkretisieren.

wurden bewusst mehrere Dienste in der Backend-Schicht mit zahlreichen Kommunikationsbeziehungen notiert, um auf den Umstand aufmerksam zu machen, dass mit steigender Zahl der im Modell berücksichtigten Komponenten die Übersichtlichkeit u. U. leidet.

Abbildung 5-13: Gesamtsicht eines Informationsmodells



Quelle: Eigene Darstellung

Es gibt keine strikte Regel, um dieses Dilemma zu vermeiden. Es kann lediglich die Empfehlung ausgesprochen werden, das Diagramm auf die Elemente zu reduzieren, die bei der Betrachtung einer konkreten Fragestellung tatsächlich relevant sind. Ist im Beispiel der nachfolgenden Abbildung etwa der Druckserver nicht von zentraler Bedeutung, weil er nicht im Fokus der Kapazitätsplanung steht, so könnte er im Diagramm auch weggelassen werden. Für andere Komponenten liegt die konkrete Umsetzung in der Notation im Ermessen des Erstellers, so dass hierzu keine verbindliche Vorgabe definiert werden kann. So sind im Beispiel die Datenbank-Server als passive Komponente vom Typ „Speicher" notiert. Es könnte hier alternativ auch eine aktive Komponten, also ein „Akteur", notiert werden, der mit den übrigen Komponenten über „Kanäle" kommuniziert.

Unabhängig davon, welcher Weg bei einer konkreten Konzeption und Implementierung gewählt wird, bieten die Sprachelemente der FMC Blockdiagramme eine Basis, um die entsprechende Systemlandschaft darzustellen und so für IT-Systemtechniker und Administratoren eine gemeinsame Grundlage zur Verständigung zu schaffen.

5.4.4 Weitere Verwendung des Informationsmodells

Ein mittels des zuvor beschriebenen methodischen Vorgehens aufgestelltes und als FMC Blockdiagramm notiertes Informationsmodell beschreibt als *funktionale Sicht* den Aufbau einer gewünschten Zielinfrastruktur, gibt Administratoren also konkrete Hinweise zu ihrer Architektur. Diese ist nun allerdings noch dahingehend zu parametrisieren, dass Anzahl und Ausstattung (mit Ressourcen wie Prozessoren, Netzwerkbandbreiten, Arbeitsspeicher, Storage) festgelegt werden, um zu einer *quantitativen Sicht* zu gelangen.

Dabei steht die zu bewältigende Kapazität (Anzahl an Endanwendern und von diesen betriebene Sitzungen, Anzahl und Ressourcenanforderungen der in den Sitzungen ausgeführten Prozesse und Applikationen) im Fokus. Es ist hier jeweils zu ermitteln, ob das System den erwarteten Anforderungen genügt und diese auch bei veränderter Last und/oder Topologie erfüllen kann. Zur Beantwortung der Fragen, wie sich ein gegebenes System bei einer erwarteten Last verhält bzw. wie ein System verändert und erweitert werden muss, um eine bestimmte Last in der Zukunft zu bewältigen, stehen mit der Simulation oder einem analytischen Herangehen verschiedene Methoden zur Verfügung.

5.4.4.1 Simulation

Ein möglicher Ansatz ist die Simulation im Sinne einer konkreten Installation von Hard- und Software und der tatsächlichen Ausführung von Arbeitslasten, d. h. der experimentelle Aufbau eines herunterskalierten Prototypen des zu untersuchenden Systems und die Simulation durch Ausführen von für die tatsächliche Nutzung typischen Aufgaben. Während in sehr kleinen Umgebungen die Last noch simuliert werden kann, indem einige Anwender ihre typische Arbeit auf dem System verrichten, müssten in größeren Umgebungen dazu detaillierte Software-Programme entwickelt werden, welche die gewünschte Last abbilden und so Rückschlüsse auf die mögliche Kapazität des Zielsystems erlauben. Durch Messung des

Prototypverhaltens könnten dann Aussagen über das spätere Verhalten der zu implementierenden Infrastruktur getroffen werden (vgl. [Menascé, 2004], S. 255).

Für Terminal Server Infrastrukturen und Desktop-Virtualisierung stehen beispielsweise mit dem (inzwischen nicht mehr weiterentwickelten) Citrix EdgeSight for Load Testing sowie Login VSI (vgl. Kapitel 12.3) bereits Werkzeuge zur Verfügung, die auf diese Aufgabe ausgerichtet sind, so dass hierfür keine Software mehr von Grund auf neu entwickelt werden muss. Es bleibt jedoch die Herausforderung, die gewünschte Last so genau wie möglich abzubilden, d. h. zu skriptieren und vor allem, den Prototypen aufzubauen. Dies führt i. d. R. zu einer hohen Genauigkeit, lässt den Ansatz je nach Größe des zu untersuchenden Systems jedoch arbeitsintensiv und teuer werden und ist somit nicht immer ein gangbarer Weg.

5.4.4.2 Analytik

Quantitative mathematische Modelle und Methoden werden im Feld der Operational bzw. Operations Research (OR) bereits seit dem frühen 20. Jahrhundert zur Unterstützung von Entscheidungsprozessen eingesetzt. Nach Ursprüngen im militärischen Sektor haben entsprechende Techniken Eingang in viele Bereiche von wirtschafts- sowie ingenieurwissenschaftlichen Anwendungen gefunden (vgl. [Kleinrock, 1975], S. xi/xii und [Lazowska, 1984], S. 2/3). Als Beispiel seien Planung von Fabriken und Produktionstechnik, Verkehrsleitsysteme und Forschungsfelder bis hin zu Teilchenphysik und Astronomie genannt. Auch im Bereich der Informationstechnik wurden frühzeitig entsprechende Ansätze verfolgt, so z. B. bei der Planung von Telekommunikationsnetzen, wie bereits dem „Plain Old Telephone System (POTS)“, also analoger Telefonie, oder auch später im Fall von paketvermittelnden Digital-Netzen, wie Integrated Services Digital Network (ISDN) oder Asynchronous Transfer Mode (ATM).

Die Modellierung unterstützt in diesen Anwendungsbereichen die Konzeption und Weiterentwicklung der Systeme und ist nützlich, in vielen Fällen sogar unumgänglich, um deren Komplexität und die Zusammenhänge der Komponenten zu erfassen und einer analytischen Herangehensweise zugänglich zu machen.

Als Alternative zum Experiment durch Prototypenbau und Simulation stehen entsprechend analytische Methoden zur Verfügung, um die Zusammenhänge zwischen der Architektur eines Systems und der zu bewältigenden Arbeitslast abzubilden. Das Herangehen (Formel 5-1) basiert darauf, ein theoretisches System-Modell S zu erstellen, auf das ein zu definierendes Last-Modell L anwendet wird, um zu einer Aussage über die daraus resultierenden langfristigen Performancewerte P zu gelangen.

Formel 5-1

$$L \times S \rightarrow P$$

Der Vorteil eines solchen analytischen Modells besteht darin, dass die Kosten gegenüber dem Experiment niedriger sind. Das Modell ist einfach zu verändern, wodurch verschiedene Entwicklungen und Handlungsalternativen ohne großen Aufwand untersucht werden können.

„Zukünftige Systeme können mit Hilfe von Modellen schon während der Planungs-, Entwurfs- und Entwicklungsphase untersucht werden." (vgl. [Müller-C., 2012], S. 13) Hierbei ist darauf zu achten, dass der Detaillierungsgrad des analytischen Modells weder zu fein noch zu grob gewählt wird. Das Modell sollte also vom zu untersuchenden System soweit abstrahieren wie nötig aber nur soweit, wie dies möglich ist, um noch zu einem zur Fragestellung passenden Ergebnis zu gelangen. Der Nachteil gegenüber dem Experiment besteht, der Abstraktion geschuldet, in einer geringeren Genauigkeit in der Aussage bezüglich der zu erwartenden zukünftigen Kapazität, denn die „Modellbildung und insbesondere die Parametrisierung von Modellen wird sich in der Regel auf Messungen oder gute Schätzungen aufgrund von Erfahrungen abstützen müssen." (vgl. [Müller-C., 2012], S. 13) Dies trifft auch und insbesondere für komplett neu zu etablierende Infrastrukturen zu, wenn noch keine belastbaren Erfahrungswerte vorliegen. Um diese zu gewinnen, kann die Simulation auf einem Prototyp in kleinem Maßstab ein Hilfsmittel sein, um erste Messwerte als Basis zu gewinnen und diese dann mittels Modellierung auf einen größeren Maßstab und umfangreichere Einsatzszenarien zu skalieren.

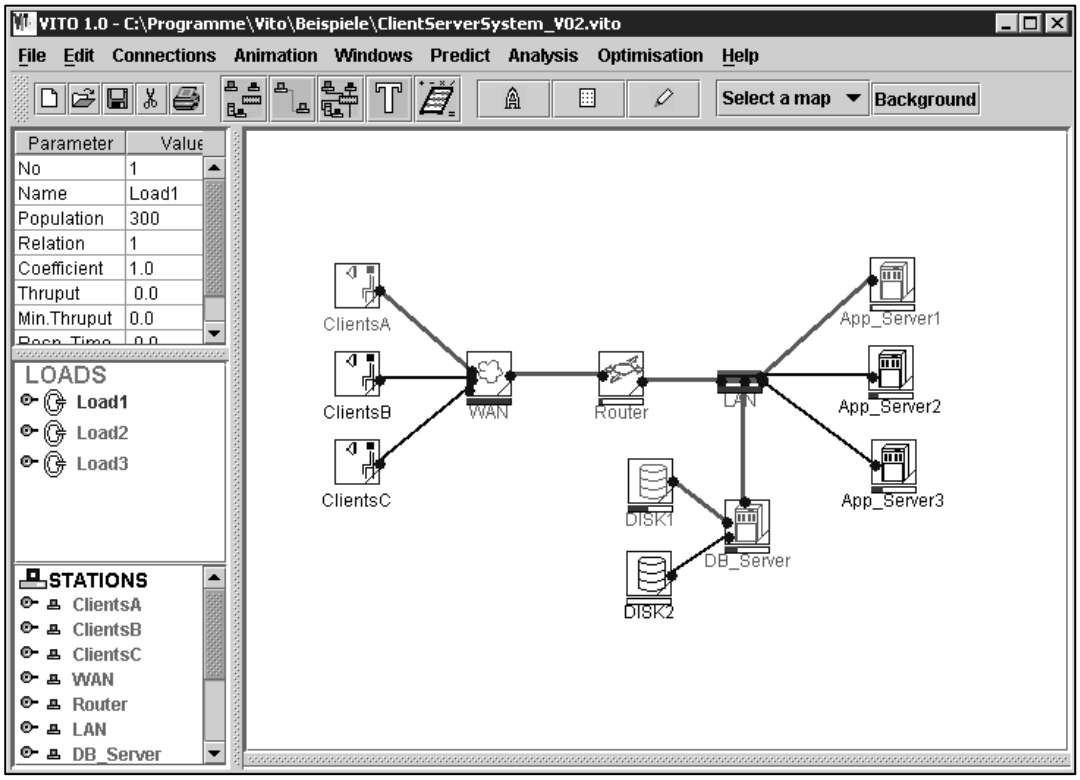
Die letzten und wichtigen Schritte im Prozess der Modellierung sind schlussendlich die finale Validierung, d. h. die Überprüfung, ob die tatsächliche Kapazität eines Zielsystems mit der durch das Modell vorhergesagten übereinstimmt, sowie – falls dies nicht zutreffen sollte – die Rückkehr zu einem früheren Schritt im Prozess und die Anpassung und Verbesserung des Modells (vgl. [Menascé, 2004], S. 257/258). Die Modellierung sollte entsprechend als iterativer Prozess begriffen werden, in dessen Verlauf durch Annährung von Modell und tatsächlichem System ein tieferes Verständnis über dessen Komponenten und ihre Zusammenhänge gewonnen werden kann.

Eine Methodik auf Basis theoretischer Modelle, welche hilft, auch komplexere Terminal Server Infrastrukturen abzubilden und im Hinblick auf zuvor gesetzte Ziele des Kapazitäts-managements zu bewerten, existierte bislang noch nicht, welshalb im Kontext dieser Arbeit ein Ansatz hierzu entworfen und erprobt wurde. Der verwendete analytische Modellierungsansatz basiert auf den sogenannten Markov-Ketten. Dabei handelt es sich um eine spezielle Form gerichteter Graphen, welche die diskreten Zustände, in denen sich ein zu modellierendes Systemen befinden kann, als Knoten abbilden, während die möglichen Übergänge zwischen diesen Knoten durch Zufallsvariable dargestellt werden (vgl. Kapitel 11.1).

Aufbauend auf dem analytischen Ansatz von Markov-Ketten und -Prozessen lassen sich Warteschlangen(netze) bilden, die nach ihrer Art als offene oder geschlossene Netze klassifiziert werden können (vgl. Kapitel 11.2.1) und der *Lösung* durch effiziente Algorithmen, wie „Mean Value Analysis (MVA)" (vgl. Kapitel 11.2.4), zugänglich sind. Diese *Lösung* besteht in einer stationären Betrachtung, d. h. einer Auswertung im Hinblick auf die langfristig stabilen Zustandswahrscheinlichkeiten – im Englischen *steady state probabilities* – des zu untersuchenden Systems.

Entsprechend fanden beim in Kapitel 6 beschriebenen Einsatz des Referenzmodells geschlossene Warteschlangennetze zur Abbildung einer SBC Infrastruktur Verwendung. Zur Unterstützung dieses Ansatzes wurde die Software VITO[100] genutzt, mit der auch umfangreiche Modelle mit vielen Stationen und verschiedenen Lastketten effizient lösbar sind. Das Werkzeug unterstützt damit System- und Netzwerk-Techniker beim Kapazitätsmanagement, wobei die Modellierung sich an Anwendungen orientiert, wie sie auch für das Zeichnen von Netzwerkdiagrammen zum Einsatz kommen. Als Vorlage hierfür dient ein zuvor aufgestelltes FMC Blockdiagramm, welches in ein VITO-Modell überführt und dann parametrisiert wird. Komponenten, wie z. B. Clients, Server, Festplatten-Subsysteme, Router, Switches oder Netzwerksegmente, werden innerhalb von VITO durch Icons repräsentiert, die zum besseren Verständnis der Abbildung durch grafische Elemente verbunden werden können (Abbildung 5-14).

Abbildung 5-14: Die Benutzeroberfläche von VITO



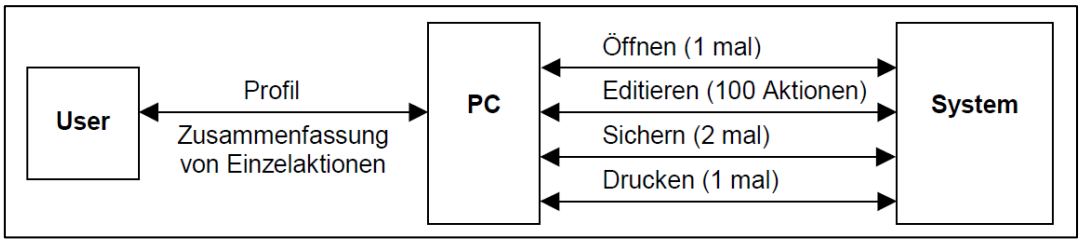
Quelle: Eigene Darstellung, Modell aus [Müller-C., 2005]

[100] VITO – A Tool for Capacity Planning and Performance Analysis of Computer Systems and Communication Networks (http://sysmod.icb.uni-due.de/index.php?id=54 (abgerufen am 01.02.2016)): „VITO entstand im Rahmen des vom BMBF geförderten Verbundprojektes MAPKIT, an welchem die Arbeitsgruppe an der Universität Essen sowie die Unternehmen Fujitsu Siemens Computers, Siemens Business Services und Materna Information and Communications beteiligt waren. In VITO wird der approximative Bard/Schweitzer-Algorithmus verwendet (…)" (vgl. [Müller-C., 2001], S. III-18).

Dies ist jedoch nicht zwingend erforderlich, da die logischen Zusammenhänge zwischen den Stationen nicht grafisch, sondern über Lastketten, im Kontext von VITO einfach als *Loads* bezeichnet, abgebildet werden. Jede Lastkette repräsentiert dabei eine geschlossene Klasse gleichartiger Anforderungen an das System. Sowohl die grundlegene Kapazität einer Komponente als auch die Anforderungen einer Lastkette an einzelne Komponenten, engl. *Service Demand*, werden in Einheiten parametrisiert, wie sie in der IT-Systemtechnik gebräuchlich sind, beispielsweise Gbit/s für den Durchsatz eines LAN-Segments oder eines Routers oder Mbyte/s für Transferraten einer Festplatte.

Die logische Verknüpfung von Lastketten und Stationen erfolgt über die sogenannten *Mappings*, mittels derer die konkreten *Service Demands* einer Lastkette an alle involvierten Stationen parametrisiert werden. Allgemeine Hinweise zur Verwendung von VITO sowie Beispiele für Modelle aus der IT-Systemtechnik finden sich in [Müller-C., 2001], Kap. 2 (S. III-10ff.) sowie in [Müller-C., 2005]. Ein Basismodell mitsamt zugehöriger Arbeitslast für einen konkreten Anwendungsfall mit Bezug zum Einsatz von Terminal Servern wurde bereits in [Müller-C., 2001], Kap. 3 (S. III-21ff.) erstmals erwähnt – allerdings explizit ohne den Anspruch, ein realistisches Modell einer existierenden Terminal Server Implementierung abzubilden. Nichtsdestotrotz eignet sich das Modell als weiteres Beispiel, um mit der Modellierung in VITO vertraut zu werden und diese in Relation zum Server Based Computing zu setzen. Die Autoren wählten in diesem Fall einen sehr hohen Detaillierungsgrad, indem sie mit einem Benutzerprofil[101] das Arbeitsverhalten einzelner Benutzer und die daraus folgende Last auf einem Terminal Server sehr feinteilig beschrieben. Als konkreter Anwendungsfall diente in diesem Beispiel die Arbeit von Anwendern mit einer Textverarbeitung, die sich im Modell als repetitive Ausführung bestimmter Dialogschritte darstellt (Abbildung 5-15). Demzufolge umfasst ein typischer Arbeitsablauf das initiale Öffnen eines Dokuments, diverse Bearbeitungsschritte, mehrfaches Abspeichern der Arbeit sowie einen Ausdruck, gefolgt von der Wiederholung dieser Folge von Arbeitsschritten.

Abbildung 5-15: Benutzerprofil „Textverarbeitung“



Quelle: [Müller-C., 2001], S. III-24

[101] In diesem Kontext bezeichnet der Term „Benutzerprofil“ eine Einheit zusammengehöriger Dialogschritte, die gemeinsam als typische Last modelliert werden sollen. Der Term ist damit zu unterscheiden vom gängigen Begriff des „Benutzerprofils“ unter Windows Betriebssystemen, wie es in den Kapiteln 2.3.3 und 3.3.1.1 erwähnt wird. Letzteres bezeichnet den benutzerspezifischen Teil der Registrierung und sämtliche vordefinierten Pfade im Dateisystem, welche die Konfigurationsdaten eines Anwenders enthalten.

Über das Mapping eines solchen Benutzerprofils auf einen Terminal Server und weitere Ressourcen, wie Datei- und Druck-Server, ließ sich die Ist-Situation dieses Anwendungsfalls abbilden und eine Prognose über die zukünftige Kapazität des Gesamtsystems ableiten. Auf diese Weise konnte die zu erwartende exponentielle Zunahme der Antwortzeiten bei steigender Anzahl der Benutzeraktionen, sprich Ausführungen des Benutzerprofils, prognostiziert werden (vgl. [Müller-C., 2001], S. III-30) – ein weiteres Beispiel für den auch von Menascé et al. beschriebenen und im Englischen als *Thrashing* bezeichneten Effekt (vgl. Kapitel 4.1.2 sowie [Menascé, 2004], S. 15). Dies sollte entsprechend die Ausgangsposition für den Aufbau eines Kapazitätsmanagements bilden.

Zwar folgt daraus für die Analyse genau dieses zu untersuchenden Benutzerprofils ein valides Modell. Bei dem Versuch, einen solchen Ansatz auf das Beispielszenario mit mehreren Terminal Servern, einer Virtualisierungsschicht und mehreren Netzsegmenten zu übertragen, wird jedoch offenbar, dass ein höherer Grad der Abstraktion notwendig ist, um die deutlich komplexere Infrastruktur erfassen zu können. Etwa für mehr als 80 Anwendungen einzelne Benutzerprofile erstellen zu wollen, würde zum einen detaillierte Kenntnisse über das typische Verhalten der Anwender *pro Applikation* voraussetzen und zum anderen das Modell kaum mehr beherrschbar machen. Die Auswertung des Ist-Standes und insbesondere Anpassungen am Modell würden erschwert, weshalb im Rahmen des im folgenden Kapitel beschriebenen Einsatzszenarios ein höherer Abstraktionsgrad gewählt wurde.

5.5 Herausforderungen der Referenzmodellierung

Die vorangegangenen Ausführungen zu Ansätzen der Referenzmodellierung lassen zu erwartende Schwierigkeiten bzw. besondere Herausforderungen bei der Konstruktion und Verwendung von Referenzmodellen erkennen. Diese beginnen bereits in der Analysephase einer zu untersuchenden Problemstellung, wo „Ursachen (…) in den Schwierigkeiten der objektorientierten Strukturierung des Gegenstands liegen (…) Ein profundes Wissen über relevante Eigenschaften und Verhaltensweisen von Objekten des Anwendungsbereichs ist erforderlich, um eine adäquate Klassenbildung vorzunehmen." (vgl. [Brocke, 2003], S. 140). Neben der Kenntnis der Methoden, Werkzeuge und Sprachen der Referenzmodellierung ist also ein fundiertes Fachwissen über die technischen und organisatorischen Eigenschaften des zu modellierenden Sachverhalts unabdingbar.

Bei Auswahl und Anwendung einer Referenzmodellierungssprache stellt sich die Frage nach möglicher Automatisierung und maschineller Verarbeitung der in der Sprache formulierten Zusammenhänge. „Die Formalisierung einer Modellierungssprache bewegt sich in dem Spannungsfeld zwischen theoretischer Präzision und pragmatischer Handhabbarkeit: Einerseits ist eine Formalisierung der Modellierungssprache notwendig, um sie maschinell verarbeiten zu können. Andererseits müssen formale Konzepte auch realweltlich gedeutet werden, da eine vollständige Formalisierung der Sprache im Sinne eines mathematischen Kalküls keine Aussagen über die reale Welt liefert. (…) Folglich erscheint es vielversprechend, nach Ansätzen zu suchen, wie natürlichsprachliche Repräsentationen nahtlos in vollständig formalisierte Referenzmodellrepräsentationen überführt werden können." (vgl. [Fettke, 2004], S.16).

Bezogen auf die Modellierung von SBC Infrastrukturen und die zuvor ausgewählte semi-formale Sprache der FMC lässt sich daraus die Frage ableiten, wie modellierte Sachverhalte syntaktisch und semantisch auf ihre Korrektheit hin überprüft werden können. Im Rahmen dieser Arbeit wurden die Schritte vom Referenzmodell zu einem konkreten in FMC notierten Informationsmodell, von dort weiter zu einem in VITO modellierten Warteschlangennetzwerk und schließlich vom theoretischen Modell zu einer praktischen Implementierung manuell ausgeführt. Eine mögliche Automatisierung dieses Prozesses wird Gegenstand zukünftiger Forschungsfragen sein (vgl. Kapitel 7.2).

Eine weitere Herausforderung besteht in der Frage, wie der Einsatz von Methoden der Referenzmodellierung quantativ und qualitativ zu bewerten ist bzw. welche bewerteten Größen hierzu herangezogen werden können. „Vor der Implementierung ist eine (…) empirische Überprüfung naturgemäß nicht möglich, nach der Implementierung sind bereits Investitionen getätigt, die man durch den Einsatz von Referenzmodellen positiv beeinflussen will. (…) Problematisch ist in diesem Zusammenhang (…) das Konzept des Benchmarkings einzustufen. Zwar verlangt eine sinnvolle Referenzmodellierung eine Menge vergleichbarer Prozesse in verschiedenen Unternehmen. (…) Ein Rückgriff auf konkretes Datenmaterial anderer Unternehmen ist folglich immer mit einer hohen Fehlerwahrscheinlichkeit verbunden." (vgl. [Remmert, 2001], S. 55/56)

Im Hinblick auf SBC Infrastrukturen ist eine umfangreichere Datenbasis aus dem Einsatz des Referenzmodells in vielen verschiedenen Einsatzszenarien noch nicht vorhanden und es wird Gegenstand weiterer Forschung sein, die im Rahmen dieser Arbeit entwickelte Methodik in zukünftigen Projekten einzusetzen und Vergleiche zwischen diesen Projekten anzustellen (vgl. Kapitel 7.2). Die Validierung des Referenzmodells im folgenden Kapitel zeigt an Hand eines Beispielszenarios, dass sich für Planung und Implementierung einer SBC Infrastruktur durch den Einsatz des Modells deutliche Vorteile ergeben. Für die zukünftige Verwendung lässt sich daraus ableiten, dass das Referenzmodell „zum einen als Instrument zur Evaluierung und Verbesserung einer bestehenden Situation dienen" (vgl. [Forster, 2013], S. 34) kann, auch wenn es erst zur Weiterentwicklung einer schon vorhandenen Infrastruktur verwendet wird und diese Infrastruktur ursprünglich noch ohne Kenntnis des Referenzmodells entwickelt worden ist. Zum anderen bietet das Referenzmodell eine wertvolle Konstruktionshilfe, wenn eine Infrastruktur von Grund auf neu konzipiert und aufgebaut werden soll.

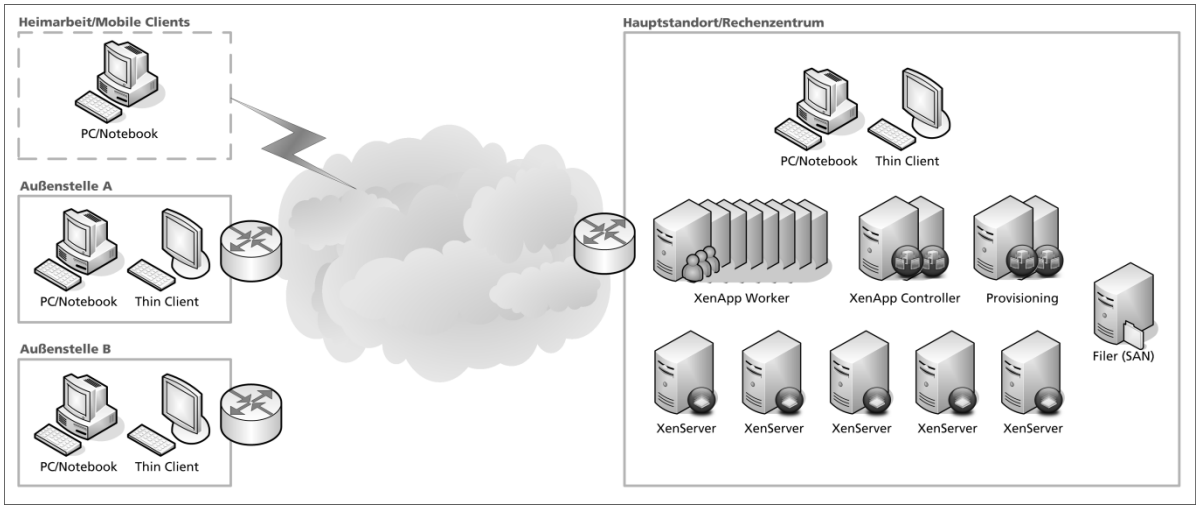
6 Validierung des Referenzmodells

Die in den vorherigen Kapiteln dargelegten Überlegungen zu dem Referenzmodell für SBC Infrastrukturen und daraus abgeleiteten Informationsmodellen werden im Folgenden anhand eines konkreten Beispielszenarios erläutert. Dazu wird ausgehend von einem praktischen Anwendungsfall ein Informationsmodell erstellt und zunächst mittels analytischer Methoden untersucht. Daraufhin erfolgt die Implementierung einer SBC Infrastruktur, in der schließlich mittels mehrerer Monitoring- und Statistik-Werkzeuge über mehrere Monate Daten zum Betrieb gesammelt werden. So wird dann das Modell validiert bzw. in einem weiteren Durchlauf feiner justiert. Das Vorgehen entspricht damit dem in Kapitel 4.3 beschriebenen und um den Ansatz der Referenzmodellierung (Abbildung 5-5) ergänzten Lebenszyklus.

6.1 Ausgangssituation

Ein Unternehmen mit 600 Mitarbeitern betreibt an seinem Hauptstandort ein Rechenzentrum, um die Mitarbeiter mit IT-Dienstleistungen zu versorgen. Von der Belegschaft sind 500 Mitarbeiter am Hauptstandort beschäftigt. Durch Übernahmen kommen nun weitere 70 Mitarbeiter in einer Außenstelle A und zusätzlich 30 Mitarbeiter in einer weiteren, kleineren Außenstelle B hinzu. Die Datenverarbeitung erfolgte bislang mit nur einem Standort lokal auf den Client-Computern nach dem Prinzip der entfernten Datenhaltung. Um den Bedarf für die lokale Datenverarbeitung auf den Clients zu minimieren und den Aufbau dezentraler Server-Ressourcen in den Außenstellen zu vermeiden, sollen zusätzlich zu den herkömmlichen Desktop-PC auch Thin Clients in Verbindung mit Terminal Servern zum Einsatz zu kommen.

Abbildung 6-1: Ein Beispielszenario



Quelle: Eigene Darstellung

Da im Backend bereits eine Virtualisierungsinfrastruktur basierend auf dem Hypervisor Citrix XenServer vorhanden ist, soll darauf aufbauend eine virtualisierte Terminal Server Farm unter Verwendung von Citrix XenDesktop bzw. XenApp etabliert werden.

Die Clients am Hauptstandort und in den Außenstellen sind jeweils an Ethernet LAN-Segmente angeschlossen. Die Außenstellen werden über Router und permanente, synchrone Netzwerkverbindungen über Weitverkehrsnetze angebunden (vgl. Abbildung 6-1). Auf den WAN-Strecken ist die Kommunikation mittels VPN-Tunneln abgesichert. Die Client-Netze am Hauptstandort sind über Switch-Verbindungen direkt an den Server-Backbone im Rechenzentrum angeschlossen.

Über die IT-Arbeitsplätze – Desktop-PC und Thin Clients – lokal an den Standorten hinaus sollen die Mitarbeiter weiterhin die Möglichkeit erhalten, von unternehmenseigenen Notebooks und privaten Geräten aus mobil bzw. an Heimarbeitsplätzen IT-Dienste zu nutzen. In diesem Fall erfolgt der Zugriff auf den Hauptstandort über asynchrone Verbindungen (typischerweise Asymmetric Digital Subscriber Line (ADSL)) oder aber Mobilfunk (in der Regel Universal Mobile Telecommunications System (UMTS) mit der Erweiterung High Speed Packet Access (HSPA)). Zur Vereinfachung des Modells wird jedoch als repräsentativ für einen solchen Arbeitsplatz zunächst nur die Anbindung per ADSL angenommen. Die Absicherung einer solchen Verbindung soll mittels Software-Client auf den Endgeräten per IPsec-VPN erfolgen.

6.2 Zielsetzungen

Für die Ausgangssituation sollen nun Fragen bezüglich der Konzeption, Impementierung und späteren Weiterentwicklung der Infrastruktur beantwortet werden. So plant das Unternehmen zukünftiges Wachstum und sieht vor, an den bestehenden Standorten evtl. weitere Mitarbeiter einzustellen. Entsprechend könnte die IT-Systemtechnik mit der Aufgabe konfrontiert werden, eine Zunahme von zehn bis 20 Prozent an Endgeräten und Anwendern an den vorhandenen Standorten zu bewältigen. Für das Kapazitätsmanagement lassen sich daraus folgende konkrete Ziele ableiten, die in allgemeiner Form auch auf beliebige SBC Infrastrukturen nach dem in Kapitel 5.4 vorgestellten Referenzmodell übertragbar sind:

1. Ziel der Untersuchung ist es, in einem ersten Schritt zunächst den Status Quo der Ausgangssituation abzubilden. Hierzu ist die Architektur einer SBC Infrastruktur für das gegebene Szenario zu konzipieren und auch bezüglich der benötigten Ressourcen zu dimensionieren. Es ist dann zu untersuchen, wie die Infrastruktur bei unveränderter Lastsituation auf Störungen reagiert (Ausfall einer oder mehrerer virtueller Maschinen, Ausfall eines Hypervisor-Hosts).
2. Weiterhin ist die Frage zu beantworten, wie sich die Infrastruktur verhält, wenn sich das Nutzungsprofil der bestehenden Anwender verändert. Wie verhält sich hier z. B. das Antwortverhalten von Applikationen, wenn neue Programme hinzukommen, die höhere Anforderungen an die Leistungsfähigkeit der Server stellen, als die bisher verwendeten?
3. Daraufhin ist die Frage zu untersuchen, ob die existierende Infrastruktur bei unveränderter Topologie eine Laststeigerung von 10 bis 20 Prozent tragen kann bzw.

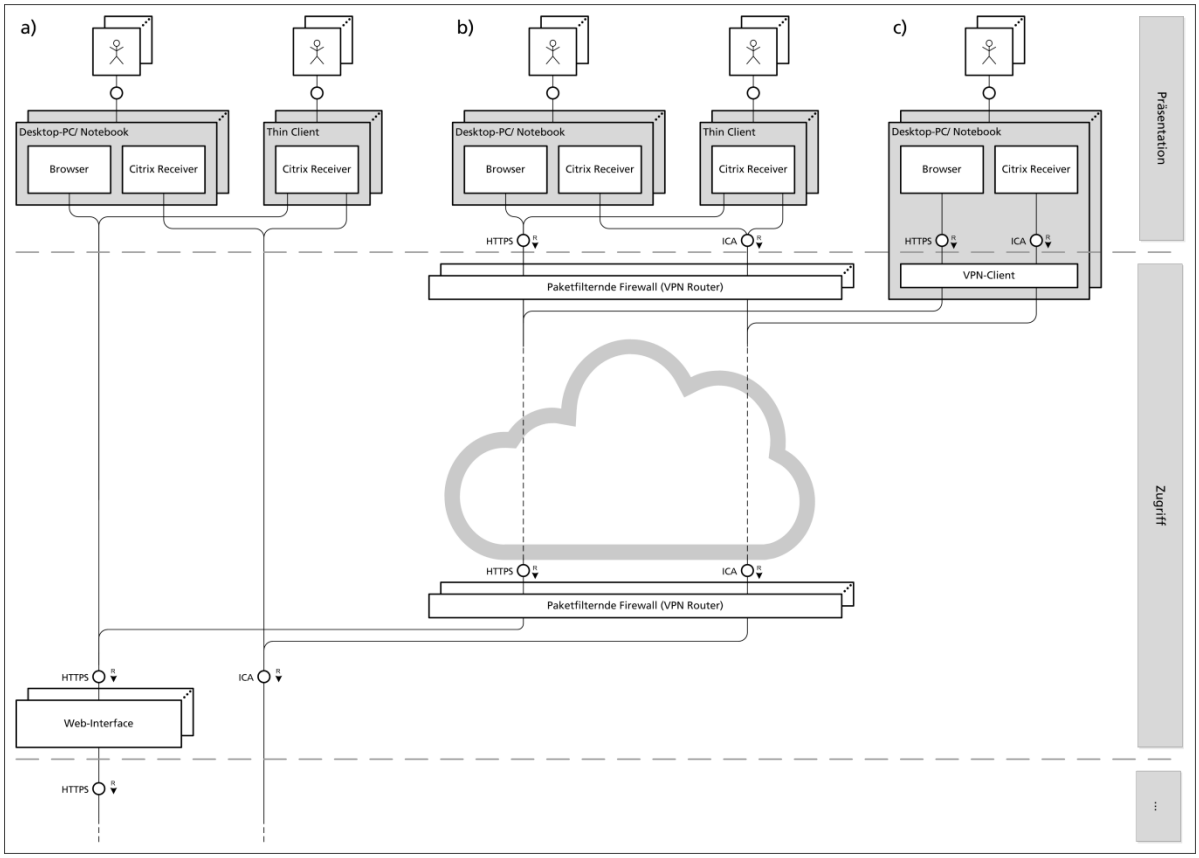
welche Änderungen vorgenommen werden müssen, um diese Last bedienen zu können.

Bevor diese Fragen im Rahmen einer analytischen Untersuchung oder konkreten Implementierung beantwortet werden können, ist in den Analyse- und Designphasen des Lebenszyklus nun zunächst die Ausgangssituation in einem konkreten Informationsmodell abzubilden.

6.3 Überführung in ein Informationsmodell

Die Abbildung des Beispielszenarios in einem Informationsmodell, welches auf dem Referenzmodell für SBC Infrastrukturen (vgl. Kapitel 5.4) basiert, erfolgt mittels der in Kapitel 5.4.3 beschriebenen Methodik. Dabei werden „top-down" die Schichten des Referenzmodells durchlaufen und die jeweiligen Leitfragen beantwortet, um zu einer detaillierten und strukturierten Gesamtsicht der SBC Infrastruktur zu gelangen.

Abbildung 6-2: FMC Notation – Präsentation und Zugriff



Quelle: Eigene Darstellung

Die Modellierung beginnt entsprechend mit der **Präsentationsschicht** und der Frage, *welche Client-Computer* zum Einsatz kommen sollen. Im vorliegenden Szenario handelt es sich um Desktop-PC und Notebooks mit dem Betriebssystem Microsoft Windows 7 sowie Thin Clients mit einem angepassten Linux-Betriebssystem. Mobile Clients, wie Tablets oder Smartphones,

spielen (noch) keine Rolle. Es darf und soll ein dediziertes Client-Programm auf den Windows-Rechnern installiert werden, um optimale Unterstützung der Funktionen der Terminal Server zu erreichen. Anwender rufen per Web-Browser über HTTPS eine Liste der für sie freigegebenen Applikationen und Desktops ab. Der Remote-Client „Citrix Receiver" baut daraufhin über das Protokoll ICA/HDX eine Verbindung zur gewünschten Ressource auf. Im Falle der Thin Clients ist davon auszugehen, dass ein entsprechender Client bereits im angepassten Linux enthalten ist. Dieser wird zentral vorkonfiguriert und erledigt direkt die Enumeration der Ressourcen per HTTPS sowie auch die Verbindungsaufnahme via ICA/HDX.

Die weiteren Fragen leiten über zur **Zugriffsschicht**. Hier ist zu untersuchen, wo sich die Clients geografisch bzw. logisch im Netzwerk befinden. Anhand der Ausgangssituation sind drei verschiedene Arten des Zugriffs (**a** bis **c**) bzw. fünf Gruppen von Benutzern und Clients identifizierbar (Abbildung 6-2).

Im einfachsten Fall (**a**) befinden sich zwei Gruppen von Desktop-PC und Thin Clients direkt am Hauptstandort des Unternehmens, also im sicheren internen Netz. Entsprechend sind beim Zugriff keine weiteren Sicherungsmaßnahmen zu ergreifen und die Clients können direkt mit dem Web-Interface bzw. den Terminal Servern in der Anwendungsschicht interagieren.

Im Fall der Zweigstellen (**b**) befinden sich wiederum jeweils zwei Gruppen von Desktop-PC und Thin Clients an den entfernten Standorten. Diese Standorte sind über permanente VPN-Tunnel mit dem Hauptstandort verbunden. Die Kommunikation zwischen diesen Standorten erfolgt mittelbar über unsichere Weitverkehrsnetze. Da jedoch die Absicherung der Kommunikation für alle Endpunkte über die paketfilternden Firewalls bzw. VPN-Router an den Standorten erfolgt, kann aus der Perspektive der Endkunden die Verbindung als sicher betrachtet werden und es sind unmittelbar auf den Client-Computern keine weiteren Sicherungsmaßnahmen erforderlich.

Im betrachteten Beispielszenario kommt (noch) keine VPN-Technologie auf Basis von SSL zum Einsatz. Entsprechend ist ein solcher Zugriff (siehe Fall b in Abbildung 5-10) nicht zu modellieren. Stattdessen wird Fall (**c**) betrachtet, in dem eine fünfte Gruppe von Desktop-PCs oder Notebooks über eine zusätzliche Softwarekomponente, den lokal installierten IPsec-VPN Client, über die unsicheren öffentlichen Netze Kontakt zum Hauptstandort aufnimmt.

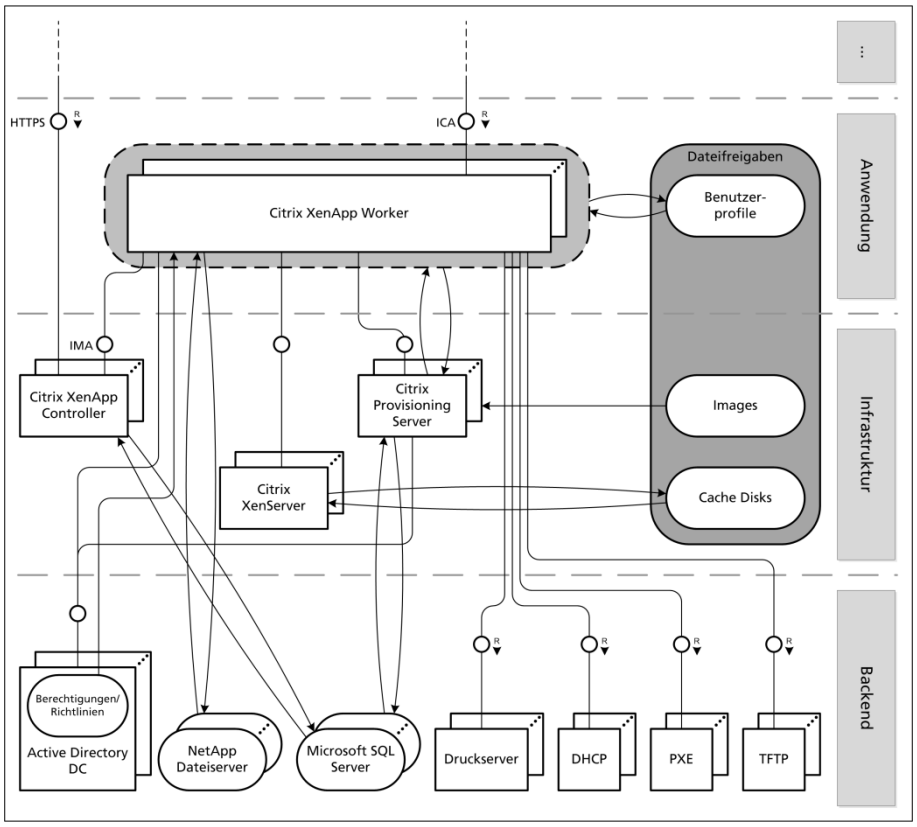
Dies sei als Beispiel dafür angeführt, dass sich mit den Sprachelementen der FMC Blockdiagramme nahezu beliebige Szenarien modellieren lassen. Je nach Zielgruppe und gewünschtem Detaillierungsgrad könnten im Modell weitere Komponenten der – im Sinne des ISO/OSI-Referenzmodells – tieferen Schichten, wie z. B. Router oder Switche modelliert werden. Darauf wurde im vorliegenden Fall zu Gunsten der Übersichtlichkeit des Diagramms jedoch verzichtet.

Beim weiteren Vorgehen gemäß der Methodik stellt sich im Hinblick auf die Anwendungsschicht des Referenzmodells die Frage, *was* die Endanwender tun wollen, also *mit welchen Applikationen* sie typischerweise arbeiten.

Im Kontext des Beispielsszenarios sind insgesamt über 80 Applikationen (32- und 64-Bit) für die Microsoft Windows Plattform relevant – darunter die Corel Graphics Suite X5 mit den Hauptanwendungen CorelDRAW und Photo-Paint, diverse Elemente der Adobe Creative Suite, das Microsoft Office Paket in der Version 2010 mit den Elementen Access, Excel, OneNote, Outlook, PowerPoint, Project, Publisher, Visio und Word, die Software OriginLab OriginPro als wissenschaftliche Anwendung im Bereich Datenanalyse und Visualisierung, als Browser aktuelle Versionen von Internet Explorer 11 und Mozilla Firefox jeweils mit Adobe Flash Plug-In sowie Clients für ein ERP-System, die als Webanwendung basierend auf der Oracle Java Runtime bzw. als native Windows-Anwendungen mit Abhängigkeit vom Oracle Database Client 11g realisiert sind.

Grundsätzlich ist damit die Entscheidung für Microsoft Windows und gegen anderweitige Plattformen, wie Linux-Distributionen oder andere UNIX-artige Systeme, auf der Serverseite vorgegeben. Alle zuvor genannten Anwendungen sind prinzipiell für den Einsatz auf Terminal Servern geeignet. Da die Anforderungen der Benutzer an die Performance auf Grund exemplarischer Messungen und Beobachtungen an vorhandenen Desktop-PCs am Hauptstandort als moderat eingestuft werden, sollen im Sinne hoher Benutzerdichte und einer effizienten Auslastung der Hardware im Rechenzentrum mehrere Terminal Server auf Basis von Citrix XenApp zum Einsatz kommen.

Abbildung 6-3: FMC Notation – Anwendung, Infrastruktur und Backend



Quelle: Eigene Darstellung

In einer aus mehreren Terminal Servern bestehenden Farm ist auf Grund der automatischen Lastverteilung im Vorhinein nicht determiniert, auf welchem der Server ein Benutzer eine Sitzung erhält. Entsprechend wichtig ist es, Maßnahmen zu implementieren, die eine konsistente Arbeitsumgebung über alle Systeme sicherstellen. Auf den Terminal Servern sollen daher im Netzwerk gespeicherte Benutzerprofile eingesetzt werden (Abbildung 6-3). Hinzu kommen über die Active Directory Domain Controller im Backend verteilte Gruppenrichtlinien, die vorgeben, dass relevante Konfigurationsdaten in diese Benutzerprofile inkludiert werden und derart restriktive Sicherheitseinstellungen greifen, dass die Anwender nur Zugriff auf ihre eigene Sitzung erhalten.

Bei der Betrachtung der Infrastruktur-Schicht steht im Fokus, *wie viele Benutzersitzungen* gleichzeitig zu unterstützen sind. Basierend auf der absoluten Anzahl an Benutzern und Endgeräten sowie auf Erfahrungswerten dazu, wie viele Benutzer typischerweise zeitgleich im Unternehmen aktiv sind, ist zunächst davon auszugehen, dass ca. 150 - bis 180 Sitzungen zu bedienen sind. Dies legt nahe, dass mehrere Terminal Server vorzusehen sind. Für ein möglichst einfaches und einheitliches Management aller Server sollen daher die Citrix Provisioning Services (PVS) für die Bereitstellung der Terminal Server aus einem gemeinsamen Basis-Image zum Einsatz kommen.

Die Anzahl der Terminal Server, welche Benutzersitzungen hosten (Citrix XenApp Worker), ist entsprechend variabel und wird daher im FMC Blockdiagramm als dynamische Struktur notiert. Die Provisioning Server beziehen die Basis-Images per CIFS-Freigabe von einem Dateiserver und stellen sie den Terminal Servern bereit. Die provisionierten Terminal Server werden als VM auf Citrix XenServer Hosts ausgeführt und speichern die über ihre Betriebsdauer veränderlichen Daten in Cache-Disks, welche ihnen über die Hypervisoren zur Verfügung gestellt werden.

Aus Gründen der Ausfallsicherheit sind in der Infrastruktur-Schicht mindestens zwei statische Instanzen von Terminal Servern (Citrix XenApp Controller) vorzusehen, die den vollständigen IMA-Protokoll-Stack implementieren und die Lastverteilung der Benutzersitzung über die übrigen Server koordinieren.

Neben den bereits genannten Active Directory Domain Controllern sind im Backend weitere Dienste erforderlich, die aus Gründen der Verfügbarkeit ebenfalls redundant ausgelegt werden sollten. Dazu zählen weitere Dateiserver zur Ablage der Nutzdaten der Endanwender, Microsoft SQL Server, welche die Terminal Server und auch die PVS zur Speicherung ihrer Konfiguration benötigen, Druckdienst und auch die für den Betrieb der PVS erforderlichen Netzwerkdienste.

Die bis zu diesem Punkt ermittelten Informationen und konzeptionellen Überlegungen führen zum FMC Blockdiagramm der geplanten SBC Infrastruktur (Abbildung 6-2, Abbildung 6-3), welches den funktionalen Aufbau der Umgebung darstellt. Das Diagramm bildet die Grundlage für eine weitere Untersuchung im Rahmen von Simulation oder Analytik, um zu einer genaueren quantitativen Spezifikation zu gelangen, also die Frage zu beantworten, wie viele Server mit welchen Ausstattungsmerkmalen letztendlich bereizustellen sind.

6.4 Analytische Untersuchung mittels VITO

Neben der Simulation im Sinne einer konkreten Installation von Hard- und Software (vgl. Kapitel 5.4.4.1) bietet sich eine analytische Untersuchung (vgl. Kapitel 5.4.4.2) des Sachverhalts an. Auf dieses Weise können bereits erste Aussagen zur Ausgestaltung der SBC Infrastruktur und ihrer Kapazität getroffen werden, ohne dass hierzu eine tatsächliche Implementierung erforderlich wäre.

Basierend auf dem im vorherigen Kapitel erstellten FMC Blockdiagramm werden im Folgenden Stationen und Lastketten (Loads) gebildet, um das Beispielszenario so in ein VITO-Modell zu überführen, dieses zu parametrisieren und im Hinblick auf die Zielsetzungen auszuwerten.

6.4.1 Stationen des VITO-Modells

Die grafische Darstellung in VITO entspricht der Topologie des Netzwerks in der zuvor in Kapitel 6.1 beschriebenen Ausgangssituation (Abbildung 6-4), wobei die Verbindungen zwischen den einzelnen Stationen lediglich der Visualisierung dienen. Die eigentliche logische Verknüpfung der Stationen erfolgt später anhand des Mappings der Lastketten.

Gegenüber dem FMC Blockdiagramm werden die Außenstellen weiter differenziert, da diese über verschiedene WAN-Verbindungen angeschlossen sind. Demgegenüber wird im Backend nur der Dateiserver in das VITO-Modell einbezogen, um die Lastketten nicht zu komplex zu gestalten. Bei der Erläuterung des Beispielszenarios sollen die Terminal Server im Vordergrund stehen, da erfahrungsgemäß die Anforderungen an die Dienste im Backend sehr gering ausfallen. Die Hilfsdienste werden i. d. R. nur aus Gründen der Ausfallsicherheit redundant ausgelegt und nicht, weil auf Grund von Leistungsanforderungen aus den höheren Schichten eine Lastverteilung über mehrere Server erforderlich wäre.

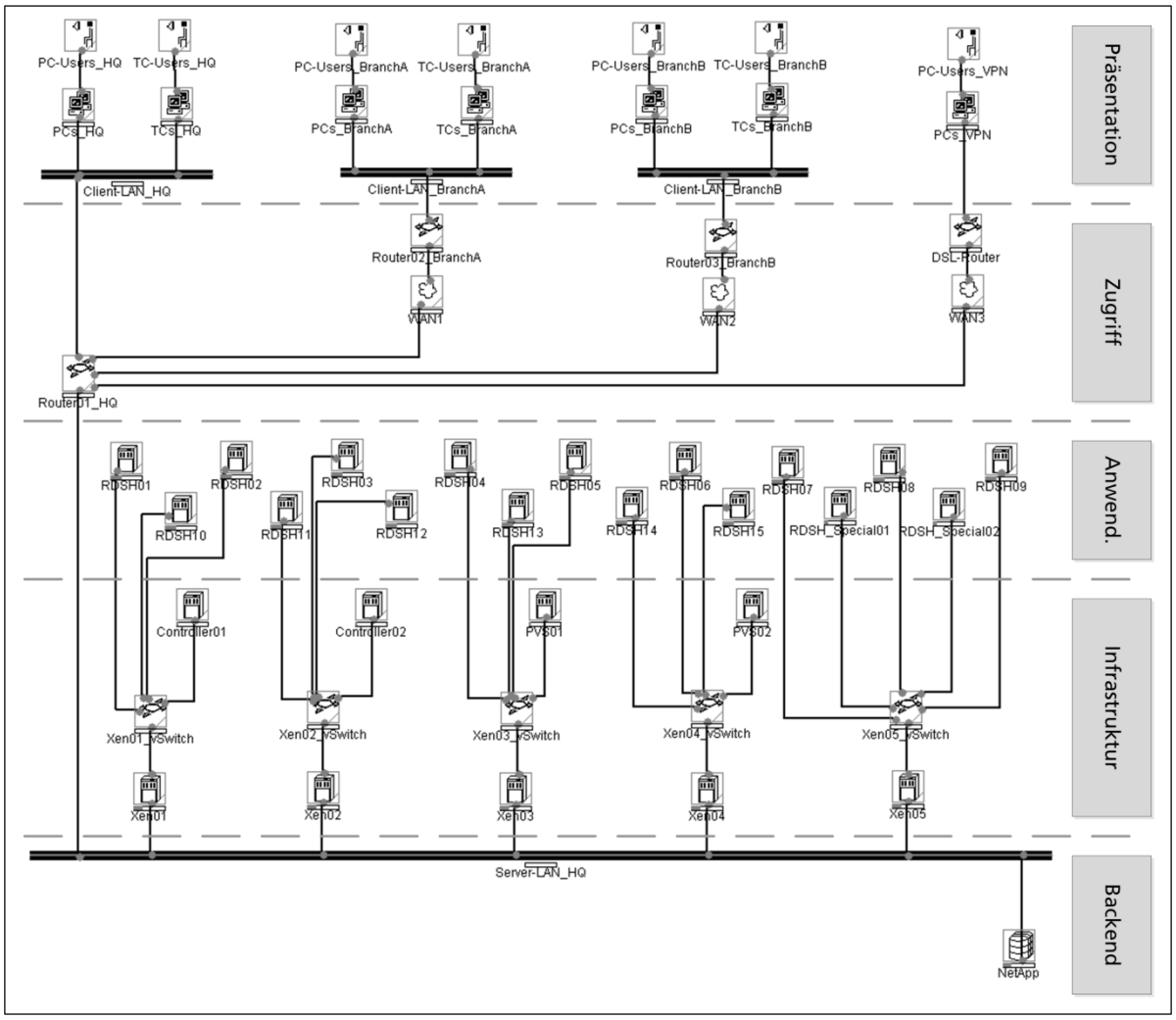
Die Gruppen der Client-Computer werden unterschieden nach Desktop-PC sowie Thin Clients und jeweils repräsentiert durch Stationen vom Typ „PC-Group". Beide Typen von Clients werden sowohl am Hauptstandort (*HQ*) als auch in den Zweigstellen A und B (*BranchA*, *BranchB*) eingesetzt. Als Bedienstrategie kommt hier der „Infinite Server (IS)" [102] zum Einsatz, da jedem Benutzer ein eigener Client zur Verfügung steht und deren Leistungsfähigkeit daher als unendlich modelliert wird. Jeder Gruppe von Clients werden jeweils Gruppen von Endanwendern mit dem Stationstyp „User" zugeordnet. Auch hier findet als Bedienstrategie IS Verwendung, um die Denkzeiten der Anwender zwischen einzelnen Arbeitsschritten zu modellieren.

Der Anwendungsfall mobiler Nutzung bzw. von Heimarbeit (*VPN*) wird exemplarisch durch eine weitere „PC-Group" von Desktop-PC und deren „User" repräsentiert. Thin Clients kommen in diesem Fall nicht zum Einsatz, so dass insgesamt sieben verschiedene Client-Gruppen und entsprechend auch mindestens sieben Lastketten den Regelbetrieb abbilden.

[102] Zu den in VITO verwendeten Bedienstrategien siehe Kapitel 11.2.2.

Wenn sich bei einer weitergehenden Analyse des Nutzerverhaltens herausstellen sollte, dass Anwender neben dem Standort auch nach Art und Umfang ihrer Bedienwünsche zu unterscheiden sind, ergeben sich unter Umständen mehrere Lastketten pro Standort und Client-Gruppe. Diese Überlegungen werden in den beiden folgenden Kapiteln konkretisiert.

Abbildung 6-4: Stationen des VITO-Modells



Quelle: Eigene Darstellung

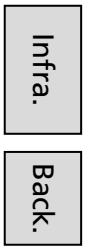
Zentraler Zugangspunkt zum Rechenzentrum ist die paketfilternde Firewall bzw. der VPN-Router des Hauptstandorts (*Router01_HQ*), der im Kontext von VITO – wie die übrigen VPN-Router (*Router02_BranchA*, *Router03_BranchB*) auch – als Station mit der Bedienstrategie „Processor Sharing (PS)" modelliert wird, da alle Clients diese Ressource gemeinsam benutzen.

Gleiches gilt auch für die Netzwerksegmente (*Server-LAN_HQ*, *Client-LAN_HQ*, *Client-LAN_BranchA*, *Client-LAN_BranchB*) und WAN-Verbindungen (*WAN1*, *WAN2*) sowie für die virtuellen Switches (*Xen01_vSwitch*, …) der XenServer Virtualisierungshosts.

Tabelle 6-1: Stationen des VITO-Modells

No.	Station	Icon	Component	Strategy	Speed	Unit
1	PC-Users_HQ	User	User	IS	-	-
2	TC-Users_HQ	User	User	IS	-	-
3	PC-Users_BranchA	User	User	IS	-	-
4	TC-Users_BranchA	User	User	IS	-	-
5	PC-Users_BranchB	User	User	IS	-	-
6	TC-Users_BranchB	User	User	IS	-	-
7	PC-Users_VPN	User	User	IS	-	-
8	PCs_HQ	PC-Group	PC-Group	IS	10.0	TA/sec
9	TCs_HQ	PC-Group	PC-Group	IS	10.0	TA/sec
10	PCs_BranchA	PC-Group	PC-Group	IS	10.0	TA/sec
11	TCs_BranchA	PC-Group	PC-Group	IS	10.0	TA/sec
12	PCs_BranchB	PC-Group	PC-Group	IS	10.0	TA/sec
13	TCs_BranchB	PC-Group	PC-Group	IS	10.0	TA/sec
14	PCs_VPN	PC-Group	PC-Group	IS	10.0	TA/sec
15	Client-LAN_HQ	Ethernet	Generic	PS	1.0	Gbits/sec
16	Client-LAN_BranchA	Ethernet	Generic	PS	1.0	Gbits/sec
17	Client-LAN_BranchB	Ethernet	Generic	PS	1.0	Gbits/sec
18	Router02_BranchA	Router	Router	PS	300.0	Mbits/sec
19	Router03_BranchB	Router	Router	PS	200.0	Mbits/sec
20	DSL-Router	Router	Router	IS	6.0	Mbits/sec
21	WAN1	WAN-Cloud	Cloud	PS	300.0	Mbits/sec
22	WAN2	WAN-Cloud	Cloud	PS	200.0	Mbits/sec
23	WAN3	WAN-Cloud	Cloud	IS	6.0	Mbits/sec
24	Router01_HQ	Router	Router	PS	300.0	Mbits/sec
25	RDSH01	Multi-Server	Multi-Server	QD4	10.0	TA/sec
(…)	(…)	(…)	(…)	(…)	(…)	(…)
39	RDSH15	Multi-Server	Multi-Server	QD4	10.0	TA/sec
40	RDSH_Special01	Multi-Server	Multi-Server	QD4	10.0	TA/sec
41	RDSH_Special02	Multi-Server	Multi-Server	QD4	10.0	TA/sec
42	PVS01	Multi-Server	Multi-Server	QD4	10.0	TA/sec
43	PVS02	Multi-Server	Multi-Server	QD4	10.0	TA/sec
44	Controller01	Multi-Server	Multi-Server	QD2	10.0	TA/sec
45	Controller02	Multi-Server	Multi-Server	QD2	10.0	TA/sec
46	Xen01_vSwitch	Router	Router	PS	10.0	Gbits/sec
47	Xen02_vSwitch	Router	Router	PS	10.0	Gbits/sec
48	Xen03_vSwitch	Router	Router	PS	10.0	Gbits/sec
49	Xen04_vSwitch	Router	Router	PS	10.0	Gbits/sec
50	Xen05_vSwitch	Router	Router	PS	10.0	Gbits/sec
51	Xen01	Multi-Server	Multi-Server	QD16	10.0	TA/sec

No.	Station	Icon	Component	Strategy	Speed	Unit
52	Xen02	Multi-Server	Multi-Server	QD16	10.0	TA/sec
53	Xen03	Multi-Server	Multi-Server	QD16	10.0	TA/sec
54	Xen04	Multi-Server	Multi-Server	QD16	10.0	TA/sec
55	Xen05	Multi-Server	Multi-Server	QD16	10.0	TA/sec
56	Server-LAN_HQ	Ethernet	Generic	PS	10.0	Gbits/sec
57	NetApp	Raid	Disk	PS	250.0	MBytes/sec



Quelle: Eigene Darstellung

Eine Sonderstellung nehmen lediglich die per VPN verbundenen Clients ein. Hier verfügt in der Regel jeder Client über einen eigenen Zugangspunkt (*DSL-Router*) und eine dedizierte WAN-Verbindung (*WAN3*). Daher kommt die Bedienstrategie „Infinite Server (IS)" zum Einsatz, um diesen Sachverhalt in VITO zu modellieren. Alle Netzwerk-Komponenten werden mit ihrer maximalen Übertragungsrate parametrisiert.

Dazu werden die Bandbreiten der jeweiligen Netzwerkverbindungen (Tabelle 6-2) verwendet, wobei alle Verbindungen jeweils als ein Link und nicht getrennt nach Up- und Downstream modelliert werden. Als Übertragungsrate für die asynchrone DSL-Verbindung wird dabei die (höhere) Downstream-Übertragungsrate von 6 Mbit/s verwendet. Dies schränkt die Aussagekraft des Modells nicht ein, da es dem Verhalten der Protokolle für die entfernte Präsentation entspricht, dass die Anforderungen an die Bandbreite im Downstream höher sind. In der Richtung vom Client zum Server sind in der Regel nur die Tastatur- und Mausbefehle zu übertragen.

Tabelle 6-2: Netzwerk-Komponenten und -segmente

Netzwerk-Komponente/-segment	Verbindungstyp	Übertragungsrate
LAN 1 – Server-Netzwerk am Hauptstandort	Switched	10 GBit/s
LAN 2 – Client-Netzwerk für PC und Notebooks am Hauptstandort	Switched	1 GBit/s
LAN 3 – Client-Netzwerk für Thin Clients am Hauptstandort	Switched	1 GBit/s
LAN 4 – Client-Netzwerk in Zweigstelle A	Switched	1 GBit/s
LAN 5 – Client-Netzwerk in Zweigstelle B	Switched	1 GBit/s
Router 1 (Hauptstandort)	Shared	300 MBit/s
Router 2 (Zweigstelle A)	Shared	200 MBit/s
Router 3 (Zweigstelle B)	Shared	100 MBit/s
WAN 1 – Verbindung Zweigstelle A zu Hauptstandort	Shared	200 MBit/s
WAN 2 – Verbindung Zweigstelle B zu Hauptstandort	Shared	100 MBit/s
WAN 3 – Verbindung Client per DSL zu WAN	Dedicated	(down) 6 MBit/s (up) 0,5 Mbit/s

Quelle: Eigene Darstellung

Im Rechenzentrum bildet ein Pool bestehend aus fünf XenServer Hypervisor Hosts die Virtualisierungsplattform für den Betrieb der Terminal Server Farm. Die XenServer nutzen per

Network File System (NFS) Storage von einem NetApp Dateiserver. Diese Verbindung wird allerdings nur für die Cache-Disks der Terminal Server benutzt, da diese mittels Citrix Provisioning Services aus gemeinsamen Basis-Images gestartet werden. Die Provisioning Server laufen ebenfalls als VM auf dem XenServer Pool und laden die Images per Common Internet File System (CIFS) vom Dateiserver. Dieser Dateiserver (*NetApp*) wird als System vom Typ „Disk" modelliert, verwendet die Bedienstrategie PS und als Geschwindigkeit seine maximale Datenrate, gemessen in MBytes/s.

Da im vorliegenden Beispielszenario noch keine Erfahrungen mit Terminal Servern vorliegen, kann die Dimensionierung der Server, das sogenannte „Sizing", in einem ersten Schritt nur auf Basis von Erfahrungswerten aus anderweitigen Projekten und darauf aufbauend begründeten Abschätzungen erfolgen. Ausgangspunkt bildet der Richtwert von vier virtuellen CPU (vCPU) und 16 GB Hauptspeicher pro Terminal Server VM[103]. Richtwerte für die zu erwartende Anzahl von Benutzern schwanken je nach Hardwareausstattung der unterliegenden Hardware und Anforderungen der Endanwender zwischen zehn und 30 Benutzern pro virtuellem Terminal Server. Eine Unterscheidung der Benutzer nach ihren Anforderungen ist insbesondere bei der Entwicklung von Modellen im Bereich der IT-Systemtechnik nicht unüblich, wo die Verwendung von mindestens zwei bis drei Klassen typischer Benutzer etwa nach *Light User*, *Medium User* oder *Power User* gebräuchlich ist (vgl. dazu auch Kapitel 11.2.4).

Im vorliegenden Fall wird mangels Erfahrung aus produktivem Betrieb zunächst ein konservativer Ansatz gewählt und das VITO-Modell auf Basis von zwölf Benutzern pro Terminal Server[104] aufgebaut. Bei einer Höchstzahl von 180 zu erwartenden Sitzungen entspricht dies 15 virtuellen Terminal Servern. Diese provisionierten XenApp Worker veröffentlichen komplette Desktop-Sitzungen sowie auch einzelne Anwendungen. Die Endanwender an Linux Thin Clients nutzen nur komplette Desktop-Sitzungen. Die Endanwender an Desktop-PC und Notebooks nutzen sowohl Desktop-Sitzungen als auch einzelne Anwendungen.

Zwei weitere Worker stellen zusätzliche Anwendungen bereit, die auf Grund von technischen oder lizenzrechtlichen Rahmenbedingungen nicht auf den übrigen Servern installiert werden können. Diese Anwendungen werden wiederum innerhalb einer Desktop-Sitzung auf den anderen Terminal Servern oder aber direkt von den PC und Notebooks aus aufgerufen. Weiterhin fungieren zwei nicht provisionierte Systeme als Controller, die auch das Citrix Webinterface hosten.

Die grafische Darstellung der Ressourcen im Server-LAN entspricht *einer möglichen Verteilung* der virtuellen Maschinen. Insgesamt laufen als Virtualisierungsplattform fünf nach ihrer

[103] Dieser Richtwert kann sowohl für neuere Implementierungen unter XenDesktop/XenApp 7.x als auch für XenApp 6.5 zu Grunde gelegt werden, vgl. „Quick server sizing for XenApp 7+" (http://blogs.citrix.com/2014/04/01/quick-server-sizing-for-xenapp-7/ (abgerufen am 01.02.2016)).

[104] Für Systeme mit zwei physischen CPU-Sockeln kann pro Terminal Server VM eine Anzahl von 36 Benutzern mit geringen Anforderungen, 24 Benutzern mit mittleren Anforderungen oder zwölf Benutzern mit hohen Anforderungen an die Leistung des Servers angenommen werden, vgl. „What's the optimal XenApp 6.5 VM configuration?" (http://blogs.citrix.com/2013/01/07/whats-the-optimal-xenapp-6-5-vm-configuration/ (abgerufen am 01.02.2016)).

Kapazität gleichwertige XenServer Hosts (*Xen01*, …) mit jeweils zwei Prozessoren à acht Kerne. Die Systeme verwenden entsprechend die Bedienstrategie „Queue Dependent (QD)" bzw. konkret QD16. Die fünf Hosts bilden einen Pool mit Hochverfügbarkeit und Lastverteilung, so dass die Verteilung der VM über die Hosts zufällig erfolgt, wobei bei einer stationären Betrachtung des Regelbetriebs alle Hosts gleichmäßig ausgelastet werden.

Auch die virtuellen Maschinen werden gemäß der Bedienstrategie QD modelliert. Alle Terminal Server (*RDSH…*) und die Provisioning Server (*PVS…*) verfügen jeweils über vier virtuelle Prozessoren (vCPU), was QD4 entspricht. Den beiden Controllern sind jeweils lediglich zwei vCPU zugeordnet, es folgt daraus also QD2. Die Virtualisierungshosts sowie alle VM werden bezüglich ihrer Geschwindigkeit mit Transaktionen pro Sekunde (TA/s) parametrisiert. Im Kontext des Modells nimmt der Begriff der „Transaktion" zwei Bedeutungen an und beschreibt zum einen aus Sicht der Endanwender und Clients sämtliche Aktionen, die die Benutzer unmittelbar ausführen (Tastatur- und Mausbefehle), worauf ebenso unmittelbar eine Rückmeldung des Systems erwartet wird. Zum anderen sind alle Operationen zu berücksichtigen, die mittelbar aus den Tastatur- und Mausbefehlen resultieren. Dies schließt sämtliche Operationen, Berechnungen sowie weiteren Kommunikationsbeziehungen mit Servern im Backend ein, die durch die Anwendungsprogramme und Systemprozesse auf den Terminal Servern ausgelöst werden.

6.4.2 Lastketten und deren Abbildung auf die Stationen

Für die Verteilung der Sitzungen auf einzelne Server im Modell werden die Server zunächst nach Art der veröffentlichten Anwendung unterschieden und in zwei Gruppen zusammengefasst. Die größere Gruppe bilden die Server für die Standard-Anwendungen und Desktops. Die beiden Server für die Spezial-Anwendungen bilden die zweite Gruppe. Es wird angenommen, dass diese für die Auslastung der Umgebung nur eine untergeordnete Rolle spielen und maximal fünf Prozent der Sitzungen darauf entfallen, während 95 Prozent der Last von der ersten Gruppe bedient werden.

Innerhalb der Gruppen erfolgt über das Load Balancing von Citrix XenApp eine homogene Gleichverteilung der Sessions über die Server. Bei fünfzehn Standard-Servern wird eine Sitzung mit der Wahrscheinlichkeit von $1/15$ auf einen der Server geleitet, bei nur zwei Servern für Spezial-Anwendungen beträgt die Wahrscheinlichkeit entsprechend $1/2$. Es folgen die kombinierten Wahrscheinlichkeiten, dass eine Sitzungsanfrage an einen bestimmten Server geleitet wird, als Produkt der Wahrscheinlichkeiten nach Anwendungsgruppe und einzelnem Server (Tabelle 6-3).

Tabelle 6-3: Verteilung von Sessions auf Server

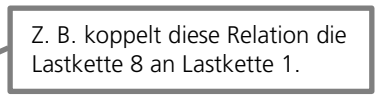
Server-Gruppe	Server	Wahrscheinlichkeit nach Anwendung	Wahrscheinlichkeit nach Server	Wahrscheinlichkeit kombiniert
Standard-Anwendungen und Desktops	RDSH01	95%	0,0667	0,0633
	RDSH02	95%	0,0667	0,0633
	RDSH03	95%	0,0667	0,0633
	RDSH04	95%	0,0667	0,0633
	RDSH05	95%	0,0667	0,0633
	(…)	(…)	(…)	(…)
	RDSH15	95%	0,0667	0,0633
Spezial-Anwendungen	RDSH_Special01	5%	0,5000	0,0250
	RDSH_Special02	5%	0,5000	0,0250
Gesamt				**100%**

Quelle: Eigene Darstellung

Für die Klassifizierung im Hinblick auf die Performance ist bei der weiteren Betrachtung des Beispielszenarios die Unterscheidung nach Art der Sitzung allein jedoch nicht zweckmäßig. Die weitere Untersuchung erfolgt daher im Folgenden auch nach geografischem Standort des jeweiligen Clients und es werden die folgenden zum Teil gekoppelten Lastketten gebildet (Tabelle 6-4).

Tabelle 6-4: Lastketten des ersten VITO-Modells

Nr.	Name	Population	Relation	Koeffizient
1	PCs_HQ_Sessions	35	1	1,0
2	TCs_HQ_Sessions	100	2	1,0
3	PCs_BranchA_Sessions	5	3	1,0
4	TCs_BranchA_Sessions	20	4	1,0
5	PCs_BranchB_Sessions	3	5	1,0
6	TCs_BranchB_Sessions	7	6	1,0
7	PCs_VPN_Sessions	10	7	1,0
8	PCs_HQ_Server-Load	35	1	1,0
9	TCs_HQ_Server-Load	100	2	1,0
10	PCs_BranchA_Server-Load	5	3	1,0
11	TCs_BranchA_Server-Load	20	4	1,0
12	PCs_BranchB_Server-Load	3	5	1,0
13	TCs_BranchB_Server-Load	7	6	1,0
14	PCs_VPN_Server-Load	10	7	1,0



Quelle: Eigene Darstellung

Diesen Lastketten liegt das Prinzip zu Grunde, dass die Kommunikation zwischen Client und Server (also die reine Übertragung von Eingabedaten und grafischer Ausgabe über das ICA-

Protokoll) auf der einen Seite sowie die Ausführung der eigentlichen Sitzungen und Applikationen auf der Server-Seite getrennt von einander modelliert werden. Die Durchsätze zweier Gruppen werden aber jeweils miteinander gekoppelt (zur Kopplung der Durchsätze nach Totzauer vgl. Kapitel 11.2.4). Jeder Gruppe von Clients, z. B. *TCs_HQ_Sessions* für die Thin Clients am Hauptstandort, wird dabei in VITO über die „Relation" die korrespondierende Arbeitslast auf dem Server, in diesem Fall *TCs_HQ_Server-Load*, zugeordnet. Der Vorteil dieses Ansatzes liegt darin, dass Antwortzeiten im Input-/Output-Dialog zwischen Client und Server sowie die Bearbeitungszeiten von Aufgaben durch die Anwendungen auf dem Server bei der Lösung und weiteren Verwendung des Modells separat ausgewertet werden können.

Vor der Lösung des Modells sind die Lastketten zunächst noch auf die Stationen abzubilden, was in VITO durch ein „Mapping" erfolgt. Es werden hier wiederum exemplarisch nur zwei Mappings – die bereits zuvor genannte Client-Gruppe (*TCs_HQ_Sessions*, Tabelle 6-5) und die zugehörige Server-Last (*TCs_HQ_Server-Load*, Tabelle 6-6) – beschrieben, da die übrigen sich prinzipiell nicht davon unterscheiden und lediglich in den benutzten Stationen abweichen.

Tabelle 6-5: Mapping der Lastkette „TCs_HQ_Sessions"

Station	Speed	Unit	Service Amount	Unit	Think-time [s]	Visit Count
Client-LAN_HQ	1,0	Gbits/sec	1,38E+03	Gbit	-	1,0000
Server-LAN_HQ	10,0	Gbits/sec	1,38E+03	Gbit	-	1,0000
Router01_HQ	300,0	Mbits/sec	0,14160156	Mbit	-	1,0000
RDSH01	10,0	TA/sec	0,10	TA	-	0,0633
(…)	(…)	(…)	(…)	(…)	(…)	(…)
RDSH15	10,0	TA/sec	0,10	TA	-	0,0633
RDSH_Special01	10,0	TA/sec	0,10	TA	-	0,0250
RDSH_Special02	10,0	TA/sec	0,10	TA	-	0,0250
PVS01	10,0	TA/sec	0,10	TA	-	0,5000
PVS02	10,0	TA/sec	0,10	TA	-	0,5000
Xen01	10,0	TA/sec	0,21	TA	-	0,2000
(…)	(…)	(…)	(…)	(…)	(…)	(…)
Xen05	10,0	TA/sec	0,21	TA	-	0,2000
Controller01	10,0	TA/sec	0,01	TA	-	0,5000
Controller02	10,0	TA/sec	0,01	TA	-	0,5000
NetApp	250,0	MBytes/sec	1,00	MByte	-	1,0000
Xen01_vSwitch	10,0	Gbits/sec	1,38E+03	Gbit	-	0,2000
(…)	(…)	(…)	(…)	(…)	(…)	(…)
Xen05_vSwitch	10,0	Gbits/sec	1,38E+03	Gbit	-	0,2000
TC-Users_HQ	-	-	-	-	5,00	1,0000
TCs_HQ	10,0	TA/sec	1,00	TA	-	1,0000

Der Bandbreitenbedarf einer Sitzung auf den Netzwerkkomponenten wird mit durchschnittlich für das ICA-Protokoll benötigten 145 Kbit/s parametrisiert.

Quelle: Eigene Darstellung

Maßgeblich für eine Client-Sitzung ist die vom ICA-Protokoll benötigte Bandbreite, welche hier mit 145 Kbit/s[105] in die Betrachtung einfließt. Diese ist auf alle von der jeweiligen Client-Gruppe benötigten Netzwerk-Komponenten und -Segmente anzuwenden. Für die Auslastung der Terminal Server und unterliegenden Virtualisierungshosts wird lediglich eine sehr geringe Grundlast für die Verwaltung der Benutzersitzung zum Ansatz gebracht. Die eigentliche Arbeitslast wird durch die an die Client-Gruppe gekoppelte Lastkette und deren Mapping repräsentiert.

Tabelle 6-6: Mapping der Lastkette „TCs_HQ_Server-Load"

Station	Speed	Unit	Service Amount	Unit	Visit Count
Server-LAN_HQ	10,0	Gbits/sec	3,23E+02	Gbit	1,0000
RDSH01	10,0	TA/sec	8,0	TA	0,0633
(…)	(…)	(…)	(…)	(…)	(…)
RDSH15	10,0	TA/sec	8,0	TA	0,0633
RDSH_Special01	10,0	TA/sec	8,0	TA	0,0250
RDSH_Special02	10,0	TA/sec	8,0	TA	0,0250
PVS01	10,0	TA/sec	0,1	TA	0,5000
PVS02	10,0	TA/sec	0,1	TA	0,5000
Xen01	10,0	TA/sec	3,0	TA	0,2000
(…)	(…)	(…)	(…)	(…)	(…)
Xen05	10,0	TA/sec	3,0	TA	0,2000
Controller01	10,0	TA/sec	0,1	TA	0,5000
Controller02	10,0	TA/sec	0,1	TA	0,5000
NetApp	250,0	MBytes/sec	1,0	MByte	1,0000
Xen01_vSwitch	10,0	Gbits/sec	3,23E+02	Gbit	0,2000
(…)	(…)	(…)	(…)	(…)	(…)
Xen05_vSwitch	10,0	Gbits/sec	3,23E+02	Gbit	0,2000

Die Besuchshäufigkeit (Visit Count) der Terminal Server wird parametrisiert mit den kombinierten Wahrscheinlichkeiten aus Tabelle 6-3.

Quelle: Eigene Darstellung

Diese Lastkette wird nur auf Ressourcen im Rechenzentrum abgebildet und ist zunächst für alle Client-Gruppen identisch parametrisiert, da noch keine signifikant unterschiedlichen Anforderungen der Anwender an den verschiedenen Standorten bekannt sind. Sollten sich diese im Rahmen des praktischen Einsatzes später ergeben, kann das Modell pro Client-Gruppe modifiziert werden.

Die Arbeitslast umfasst sämtlichen Datenverkehr im Netzwerk, der über das ICA-Protokoll für die entfernte Präsentation hinausgeht – also alle Daten, welche direkt mit dem Dateiserver

[105] Dieser Wert folgt aus Erfahrungen basierend auf im praktischen Betrieb mit dem LANrunner gemessenen Sitzungen und bewegt sich damit im Rahmen einer typischerweise zu erwartenden Spannbreite von 85-160 Kbit/s (vgl. http://blogs.citrix.com/2013/09/26/plan-ahead-with-xendesktop-bandwidth-daily-average/, abgerufen am 01.02.2016).

(*NetApp*) oder den XenServern und Provisioning Servern ausgetauscht werden. Letztere können, müssen jedoch nicht zwingend, auch am Dateiserver ankommen (vgl. Kapitel 12.1).

Für beide zuvor beschriebenen Mappings repräsentiert die Besuchshäufigkeit (*Visit Count*) die Wahrscheinlichkeiten, mit denen die einzelnen Ressourcen ausgelastet werden. Der Visit Count der Terminal Server entspricht den in Tabelle 6-3 aufgeführten kombinierten Wahrscheinlichkeiten. Analog bestimmt sich die Wahrscheinlichkeit, mit der die übrigen Komponenten verwendet werden. Bei fünf Virtualisierungshosts beträgt die Wahrscheinlichkeit, eine Sitzung zu bedienen, $^1/_5$ oder 20 Prozent. Zur Durchführung von Modellexperimenten – beispielsweise Ausfall eines Virtualisierungshosts, eines oder mehrerer Terminal Server – sind die Besuchshäufigkeiten entsprechend anzupassen.

Nachdem die Mappings für alle weiteren Lastketten analog zum zuvor beschriebenen Vorgehen parametrisiert waren, konnte das Modell in VITO gelöst werden (*Solve*). Als Ergebnis wies VITO für die angenommene Anzahl an Sessions keine zu erwartenden Engpässe auf.

Tabelle 6-7: Lösung des ersten VITO-Modells (Auszug)

Station	UtilUsr	UtilOhd	Util	Idle	Max.Util
RDSH01	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH02	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH03	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH04	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH05	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH06	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH07	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH08	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH09	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH10	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH11	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH12	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH13	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH14	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH15	44,67%	2,61%	47,28%	52,72%	60,00%
RDSH_Special01	17,63%	0,54%	18,17%	81,83%	60,00%
RDSH_Special02	17,63%	0,54%	18,17%	81,83%	60,00%

Zur Erläuterung der VITO-Ausgabe: Die Spalte „UtilUsr" (User) gibt die direkt durch den Benutzer erzeugte Prozessorlast an, die Spalte „UtilOhd" (Overhead) einen Zuschlag, der durch die Skalierung des Multiprozessorsystems hervorgerufen wird.

Die Spalte „Util" weist die kumulierte Prozessorlast aus, die Spalte „Idle" komplementär dazu den Leerlauf.

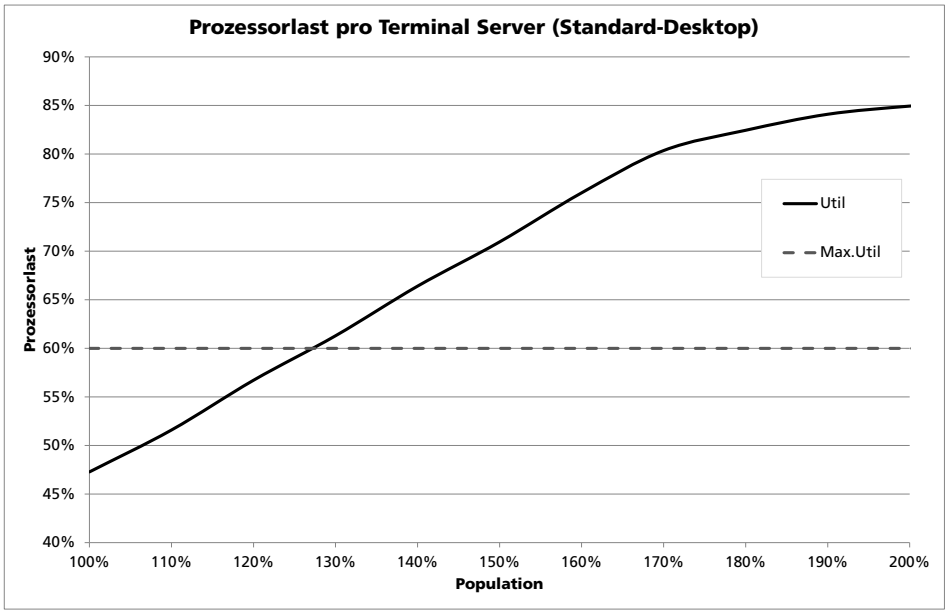
Der konfigurierbare Wert „Max. Util" markiert die Warnschwelle, bis zu der das System als benutzbar angenommen wird.

Quelle: Eigene Darstellung

Insbesondere ließ die prognostizierte Prozessorlast pro Terminal Server (Standard-Desktop) mit ca. 47 Prozent (Tabelle 6-7) eine Umgebung erwarten, die flüssiges Arbeiten im Regelbetrieb zulässt, jedoch auch genügend freie Kapazitäten bietet, um etwa den Wegfall eines Terminal Servers oder gar eines Virtualisierungshosts zu verkraften. So zeigte eine Variation der Last mit der VITO-Funktion „Predict\ Population ▸\ Vary" und einer Zunahme der Last in Schritten von

zehn Prozent in der weiteren Analyse mit der Funktion „Predict\ Population ▸\ Analysis“ bei einer Auswertung im Hinblick auf die Prozessorlast der Terminal Server, dass diese bei zunehmder Last in die Sättigung laufen (Abbildung 6-5).

Abbildung 6-5: Variation der Population in VITO



Quelle: Eigene Darstellung

Die Grenze von 60 Prozent, welche als Obergrenze für flüssiges Arbeiten in den Sitzungen auf den Servern angenommen wurde, wird erst bei einer Erhöhung der Last um ca. 30 Prozent überschritten.

6.5 Implementierung und Validierung

Nachdem die Untersuchung des VITO-Modells keine Flaschenhälse in der Leistung der SBC Infrastruktur erwarten ließ, erfolgte die tatsächliche Implementierung der Umgebung und ihr Betrieb über einen längeren Zeitraum, wobei mit den in Kapitel 12 beschriebenen Werkzeugen statistische Daten zur Auslastung der Umgebung erfasst wurden. Die Ausstattungsmerkmale und Leistungsdaten der Client-Computer sowie der physischen und virtuellen Server sind der folgenden Aufstellung zu entnehmen (Tabelle 6-8).

Tabelle 6-8: Clients und Server

Station	Leistungsdaten und Ausstattung
Client-PC	Ausprägung: Midi-Tower Prozessor: 2x Intel Xeon E5530 (2,4 GHz) Quad-Core Hauptspeicher: 8 GB Betriebssystem: Microsoft Windows 7 (SP1) Netzwerk: Physical Interface (PIF), 1 Gbit/s

Station	Leistungsdaten und Ausstattung
Thin Client	Ausprägung: IGEL UD3-LX 41 Prozessor: VIA Eden X2 U4200 (1,0 GHz) Hauptspeicher: 1 GB (Shared Memory inkl. 128 MB Grafikspeicher) Betriebssystem: IGEL Universal Desktop LX Netzwerk: Physical Interface (PIF), 1 Gbit/s
XenServer Virtualisierungshost	Ausprägung: Physischer Server vom Typ IBM BladeServer HS22 Prozessor: 2x Intel Xeon E5530 (2,4 GHz) Quad-Core Hauptspeicher: 96 GB Betriebssystem: Citrix XenServer 6.2 Netzwerk: Physical Interface (PIF), 10 Gbit/s
XenApp Controller	Ausprägung: VM Prozessor: 2x vCPU Hauptspeicher: 4 GB Betriebssystem: Microsoft Windows Server 2008 R2 (SP1) Netzwerk: Virtual Interface (VIF), 10 Gbit/s
XenApp Worker	Ausprägung: Provisionierte VM Prozessor: 4x vCPU Hauptspeicher: 12 GB Betriebssystem: Microsoft Windows Server 2008 R2 (SP1) Netzwerk: Virtual Interface (VIF), 10 Gbit/s
Provisioning Server	Ausprägung: VM Prozessor: 2x vCPU Hauptspeicher: 16 GB Betriebssystem: Microsoft Windows Server 2008 R2 (SP1) Netzwerk: Virtual Interface (VIF), 10 Gbit/s
Datei-Server	Ausprägung: Physischer Server vom Typ NetApp FAS3220 Netzwerk: Physical Interface (PIF), 10 Gbit/s

Quelle: Eigene Darstellung

Für die genauere Parametrisierung und Weiterentwicklung des VITO-Modells sind hiervon maßgeblich Rechenleistung und Netzwerkbandbreite relevant. Der Schlüssel hierzu ist eine Analyse der zur Verfügung stehenden Monitoringdaten im Hinblick auf Gruppen zusammengehöriger Datensätze mit ähnlichen Anforderungen.

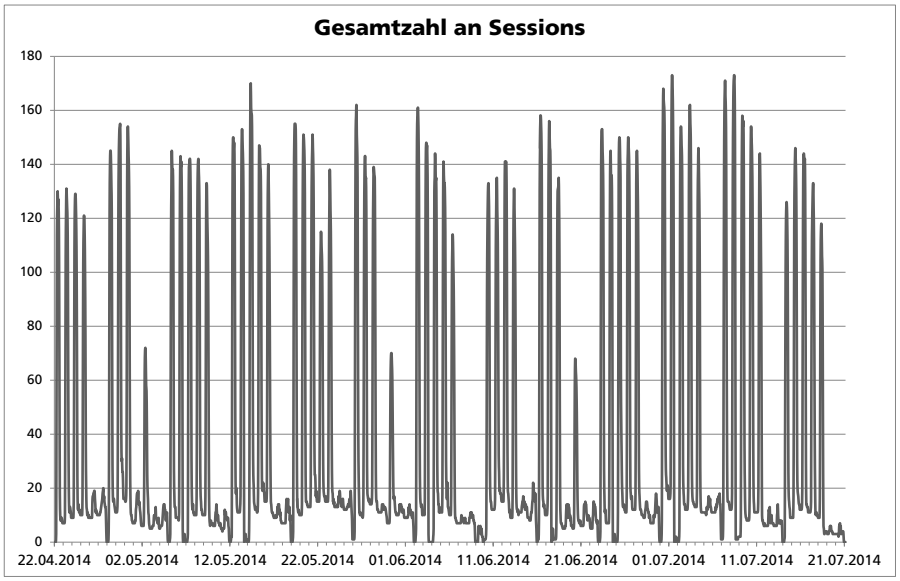
Gemäß der in Kapitel 12.4 beschriebenen Konfiguration der Monitoring-Software EdgeSight liegen in der Beispielinfrastruktur Daten der vergangenen 90 Tage vor. Diese können hinsichtlich der verschiedenen Anwendungsfälle ausgewertet werden. Mittels SQL-Abfrage (Tabelle 12-8) wird zunächst die zeitliche Verteilung der Sessions ermittelt (vgl. Abbildung 6-6).

6.5.1 Auswahl des Zeitraums für die Auswertung

Diese exemplarische Auswertung für den Zeitraum von April bis Juli 2014 zeigt, dass Wochenenden, Feiertage sowie darauf folgende Brückentage, die von Anwendern

typischerweise für Urlaub genutzt werden, sich direkt in der Nutzung der Terminal Server Infrastruktur niederschlagen – in diesem Fall erkennbar am Mai-Feiertag (01.05.), Christi Himmelfahrt (29.05.), Pfingst-Montag (09.06.) sowie Fronleichnam (19.06.).

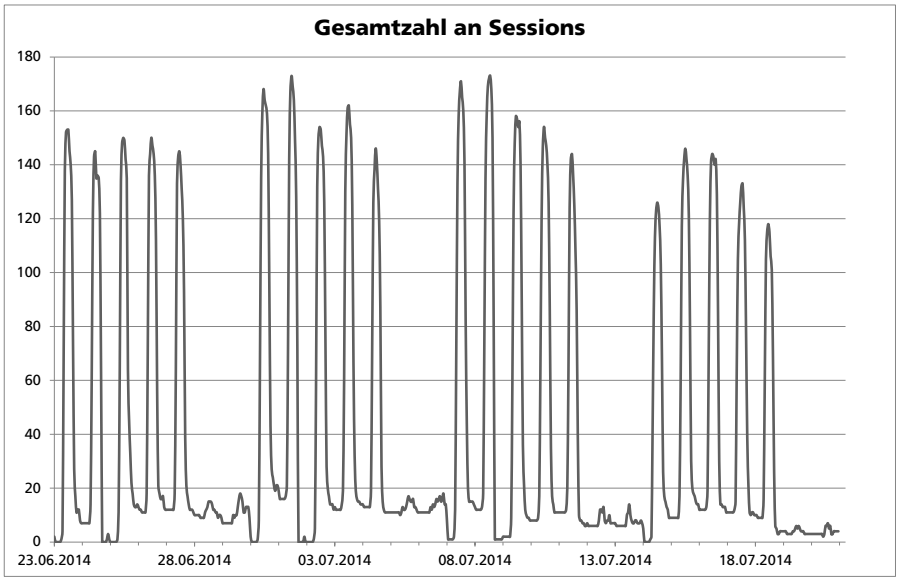
Abbildung 6-6: Gesamtzahl an Sessions



Quelle: Eigene Darstellung

Für eine repräsentative Untersuchung des Nutzungsverhaltens in einer typischen Arbeitswoche kommen entsprechend vor allem die letzten vier Wochen im Beobachtungszeitraum in Betracht (vgl. Abbildung 6-7).

Abbildung 6-7: Gesamtzahl an Sessions über vier Wochen



Quelle: Eigene Darstellung

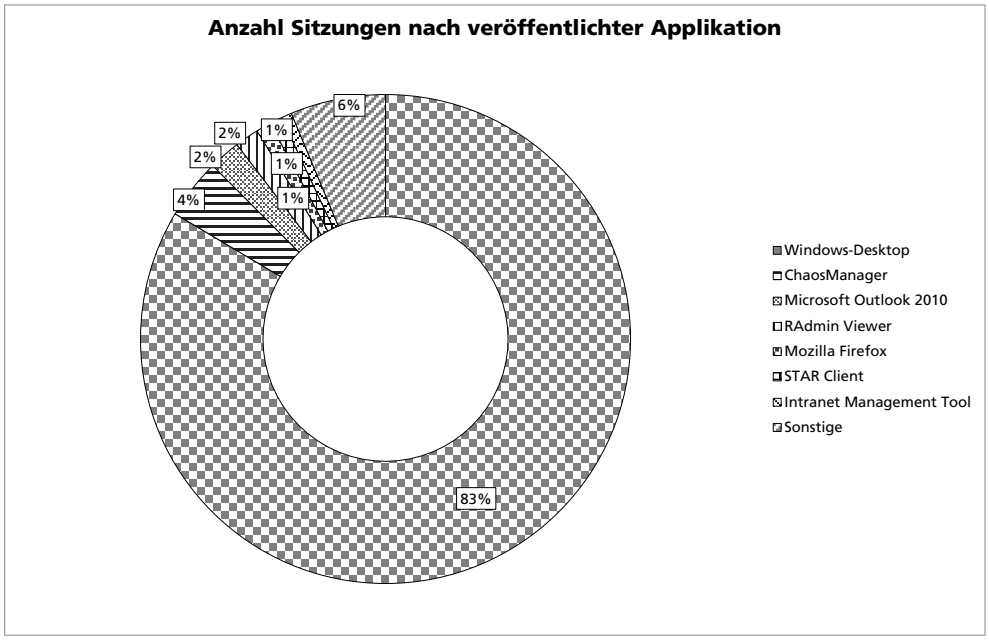
Die Einschränkung des Beobachtungszeitraums direkt bei der Auswertung der Daten ist durch eine Anpassung der SQL-Abfrage möglich, wobei der Zeitraum nicht nur auf die zuvor genannten Wochen, sondern innerhalb der Wochen auch auf Arbeitstage und innerhalb der betrachteten Arbeitstage auf die relevanten Stunden begrenzt werden kann (Tabelle 12-9). Bei der Untersuchung im Hinblick auf Klassen von Anwendungsfällen werden im Folgenden die Daten für den begrenzten Zeitraum ausgewertet und mit dem gesamten Zeitraum des Vierteljahres abgeglichen.

6.5.2 Auswertung nach veröffentlichten Anwendungen

Zunächst wird untersucht, wie die gestarteten Sitzungen nach ihrer Art verteilt sind. D. h. es ist die Frage zu stellen, welchen Anteil komplette Desktop-Sitzungen und einzelne veröffentlichte Applikationen an der Gesamtzahl aller Sitzungen haben. Die entsprechende Auswertung liefert eine Datenbankabfrage (Tabelle 12-10) in Kombination mit der bereits zuvor erwähnten zeitlichen Beschränkung (Tabelle 12-9), wobei auch die Auswahl der Tage bereits im vorherigen Abschnitt erläutert wurde. Die Begründung für die in der Abfrage vorgenommene Begrenzung auf die tägliche Arbeitszeit zwischen 10.00 und 14.00 Uhr folgt in Kapitel 6.5.3.

Im gesamten Zeitraum wurden von EdgeSight 17.823 Sitzungen erfasst. Davon entfallen mit 1.728 Sitzungen ca. zehn Prozent auf den zeitlich begrenzten Beobachtungszeitraum. Sowohl für die Gesamtzahl als auch für die Auswahl ergibt sich eine nahezu identische Verteilung auf die verschiedenen veröffentlichten Applikationen (Abbildung 6-8 zeigt die Verteilung für den eingeschränkten Zeitraum).

Abbildung 6-8: Auswertung nach veröffentlichten Anwendungen



Quelle: Eigene Darstellung

Es geht aus der Betrachtung hervor, dass die unter dem Namen „Windows-Desktop“ veröffentlichte vollständige Desktop-Umgebung mit 83 Prozent und somit deutlichem Abstand die am häufigsten genutzte Sitzung ist. Es folgen mit dem „ChaosManager“ eine selbst entwickelte Anwendung für das Verwalten von Berechtigungen auf dem Dateiserver, der E-Mail-Client Outlook sowie einige Tools und Anwendungen, die nach ihrer Nutzungshäufigkeit und auch nach ihrem im Vorhinein bereits bekannten Ressourcenbedarf jedoch keine signifikante Rolle für die Auslastung der Terminal Server spielen.

Insbesondere sind die beiden separaten Terminal Server für die Auslastung der Umgebung nicht relevant, da entsprechend einer Auswertung nach Servern **95 Prozent** der Sitzungen auf die Standard-Server entfallen und nur **5 Prozent** der Sessions von den Servern für Sonderfälle bedient werden. Dies bestätigt die Annahmen, welche der ursprünglichen Parametrisierung des Modells zu Grunde lagen. Nur im Hinblick auf die Performance könnten die dort gehosteten Anwendungen mit auf den übrigen Terminal Servern betrieben werden. Die zusätzlichen Server sind allerdings technisch notwendig, da die dort installierten Anwendungen auf Grund ihrer Anforderungen an Berechtigungen und ihrer Update-Häufigkeit nicht mit den übrigen Anwendungen gemeinsam untergebracht werden können.

6.5.3 Auswertung nach geografischem Standort des Clients

Die weitere Untersuchung im Hinblick auf die Herkunft der Benutzer und Clients erfolgt mittels einer SQL-Abfrage (Tabelle 12-11), wobei die Anzahl an Sitzungen sortiert nach Zeitstempel und IP-Subnetz des Clients zurückgegeben wird. Um die Daten zu strukturieren, können diese dann in eine Pivot-Tabelle überführt und anschließend die enthaltenen Subnetze nach geografischen Standorten gruppiert werden.

Dies wurde für das Beispielszenario zunächst über den gesamten Zeitraum durchgeführt (Tabelle 6-9 zeigt exemplarisch Auszüge aus einem Arbeitstag und einem Sonntag) und einer stichprobenhaften Prüfung unterzogen. Dabei zeigte sich zum einen, dass die Daten über den kompletten Zeitraum sehr volatil sind, was auch durch die niedrigen Mittelwerte bei sehr hoher Standardabweichung und einen Variationskoeffizienten > 1 belegbar ist.

Tabelle 6-9: Auswertung nach Standort

Timestamp	LAN 2	LAN 3	LAN 4	LAN 5	VPN	Total
(…)	(…)	(…)	(…)	(…)	(…)	(…)
16.07.2014 01:00	3	6	2	0	1	12
16.07.2014 02:00	3	6	2	0	1	12
16.07.2014 03:00	3	6	2	0	1	12
16.07.2014 04:00	3	6	2	0	1	12
16.07.2014 05:00	3	6	3	0	1	13
16.07.2014 06:00	3	8	3	0	1	15
16.07.2014 07:00	3	30	7	1	3	44
16.07.2014 08:00	11	75	16	6	6	114

In **LAN 2** befinden sich die Desktop-PC und Notebooks am Hauptstandort, in **LAN 3** die Thin Clients am Hauptstandort. **LAN 4** umfasst alle Clients in der Zweigstelle A, **LAN 5** alle in der Zweigstelle B. Im Netz **VPN** befinden sich alle Clients, die von extern per IPsec VPN zugreifen.

Timestamp	LAN 2	LAN 3	LAN 4	LAN 5	VPN	Total
16.07.2014 09:00	10	102	22	7	10	151
16.07.2014 10:00	10	115	22	8	10	165
16.07.2014 11:00	10	114	21	7	8	160
16.07.2014 12:00	9	122	20	6	6	163
16.07.2014 13:00	10	117	23	7	9	166
16.07.2014 14:00	14	113	18	6	17	168
16.07.2014 15:00	14	109	19	8	16	166
16.07.2014 16:00	13	79	20	4	15	131
16.07.2014 17:00	7	35	13	2	8	65
16.07.2014 18:00	3	11	4	0	9	27
16.07.2014 19:00	3	6	3	0	8	20
16.07.2014 20:00	2	6	3	0	6	17
16.07.2014 21:00	1	6	4	0	6	17
(…)	(…)	(…)	(…)	(…)	(…)	(…)
20.07.2014 10:00	0	1	1	0	6	8
20.07.2014 11:00	0	1	1	0	7	9
20.07.2014 12:00	0	1	1	0	9	11
20.07.2014 13:00	0	2	1	0	11	14
20.07.2014 14:00	0	2	1	0	8	11
20.07.2014 15:00	0	2	1	0	13	16
20.07.2014 16:00	0	2	1	0	11	14
20.07.2014 17:00	0	1	1	0	6	8
20.07.2014 18:00	0	1	1	0	5	7
20.07.2014 19:00	0	1	1	0	4	6
20.07.2014 20:00	0	1	1	0	5	7
(…)	(…)	(…)	(…)	(…)	(…)	(…)
Mittelwert	**6**	**28**	**5**	**1**	**9**	**50**
Standardabw.	**7,03**	**40,87**	**6,88**	**2,51**	**6,85**	**61,05**
Variationskoeffiz.	**1,14**	**1,44**	**1,37**	**1,77**	**0,78**	**1,23**

In **LAN 2** befinden sich die Desktop-PC und Notebooks am Hauptstandort, in **LAN 3** die Thin Clients am Hauptstandort. **LAN 4** umfasst alle Clients in der Zweigstelle A, **LAN 5** alle in der Zweigstelle B. Im Netz **VPN** befinden sich alle Clients, die von extern per IPsec VPN zugreifen.

Quelle: Eigene Darstellung

Zu erklären ist dies durch den Beobachtungszeitraum, der, wie bereits erwähnt, zahlreiche Feier- und Brückentage und zusätzlich natürlich Wochenenden enthält sowie weiterhin sämtliche Tages- und Nachtzeiten erfasst. Naturgemäß ist außerhalb der Kernarbeitszeiten die Nutzung der Terminal Server signifikant geringer, so dass eine Auswertung des kompletten Zeitraums die gewonnenen Erkenntnisse sehr verfälschen würde. Zum anderen war aber anhand der Daten – insbesondere für den in Kapitel 6.5.1 gewählten Zeitraum – erkennbar, dass diese an Werktagen einem immer wiederkehrenden Schema folgen und die intensivste Nutzung auf die tägliche Arbeitszeit zwischen 10.00 und 14.00 Uhr entfällt.

Tabelle 6-10: Auswertung nach Standort (eingeschränkter Zeitraum)

Timestamp	LAN 2	LAN 3	LAN 4	LAN 5	VPN	Total
(…)	(…)	(…)	(…)	(…)	(…)	(…)
15.07.2014 10:00	16	110	18	6	11	161
15.07.2014 11:00	25	117	18	5	14	179
15.07.2014 12:00	13	113	19	5	11	161
15.07.2014 13:00	16	111	19	5	14	165
15.07.2014 14:00	14	105	19	5	15	158
16.07.2014 10:00	10	115	22	8	10	165
16.07.2014 11:00	10	114	21	7	8	160
16.07.2014 12:00	9	122	20	6	6	163
16.07.2014 13:00	10	117	23	7	9	166
16.07.2014 14:00	14	113	18	6	17	168
17.07.2014 10:00	7	99	16	7	12	141
17.07.2014 11:00	10	106	19	6	14	155
17.07.2014 12:00	10	111	21	6	13	161
17.07.2014 13:00	10	102	17	5	17	151
17.07.2014 14:00	12	105	19	6	15	157
18.07.2014 10:00	14	89	19	6	18	146
18.07.2014 11:00	13	92	18	5	16	144
18.07.2014 12:00	8	96	17	5	12	138
18.07.2014 13:00	8	89	13	4	16	130
18.07.2014 14:00	8	83	13	4	18	126
(…)	(…)	(…)	(…)	(…)	(…)	(…)
Mittelwert	**18**	**112**	**20**	**6**	**16**	**172**
Standardabw.	**5,84**	**12,31**	**3,85**	**1,50**	**4,14**	**18,37**
Variationskoeffiz.	**0,32**	**0,11**	**0,20**	**0,26**	**0,25**	**0,11**

In **LAN 2** befinden sich die Desktop-PC und Notebooks am Hauptstandort, in **LAN 3** die Thin Clients am Hauptstandort. **LAN 4** umfasst alle Clients in der Zweigstelle A, **LAN 5** alle in der Zweigstelle B. Im Netz **VPN** befinden sich alle Clients, die von extern per IPsec VPN zugreifen.

Quelle: Eigene Darstellung

Wird die Auswertung entsprechend auf diese Zeiten an Werktagen beschränkt, so werden die Daten homogener, was wiederum durch die Ermittlung von Mittelwert, Standardabweichung und Variationskoeffizient gestützt wird (Tabelle 6-10). Entsprechend wird der eingeschränkte Zeitraum auch für die Untersuchung der Auslastung der Prozessoren, der (virtuellen sowie physischen) Festplatten und der übrigen Komponenten verwendet.

Bezogen auf den Hauptstandort zeigt die vorherige Auswertung, dass ca. 86 Prozent der dortigen Sessions auf Thin Clients entfallen. Da in den Zweigstellen Desktop-PC und Thin Clients nicht in separaten IP-Subnetzen untergebracht sind, ist für die Untersuchung der Verteilung der dortigen Sessions zusätzlich eine Auswertung nach den Namen der Clients nötig. Es ergibt sich daraus, dass auch in den Zweigstellen die Thin Clients bei der Nutzung der Terminal Server dominieren. 17 von 20 Sitzungen in Zweigstelle A bzw. fünf von sechs

Sitzungen in Zweigstelle B gehen von Thin Clients aus. Hier ist also eine Anpassung der Lastketten im Modell bezüglich der Anzahl an Nutzern pro Standort erforderlich.

6.5.4 Auswertungen nach Klassen von Benutzern und Prozessen

Neben der geografischen Herkunft der Anwender und Clients ist zu untersuchen, ob sich bezüglich der Bedienwünsche verschiedene Klassen von Anwendern identifizieren lassen. Dazu wird zunächst eine Erweiterung der Datenbank (Kapitel 12.5.4.3) eingeführt, um die innerhalb der Sitzungen ausgeführten Prozesse entsprechend ihrer Art nach Hintergrundprozessen und den eigentlichen Anwendungsprogrammen gruppieren zu können. Unter Zuhilfenahme dieses neuen Unterscheidungsmerkmals erfolgt dann eine Auswertung der Datenbasis im eingeschränkten Beobachtungszeitraum mittels einer SQL-Abfrage (Tabelle 12-15). Ergebnis ist die gruppierte Anzahl der I/O-Operationen pro Sekunde nach Zeitstempel und Session sowie die von den I/O-Operationen verursachte CPU-Last. Für einzelne Ergebnisspalten, wie CPU-Last oder I/O-Operationen, ist die Ausgabe der Abfrage in eine Pivot-Tabelle (Tabelle 6-11) zu überführen, wobei Zeitstempel und Sitzungsnummer als Zeilenbeschriftung und der Typ des Prozesses als Spaltenbeschriftung dienen. Die Summen der CPU-Lasten der einzelnen Prozesse bilden die Werte, so dass pro Zeiteinheit des Beobachtungszeitraums und Sitzung die Anteile des Ressourcenbedarfs nach Anwendung und Hintergrundprozess dargestellt werden.

Tabelle 6-11: CPU-Last nach Anwendungen und Systemprozessen

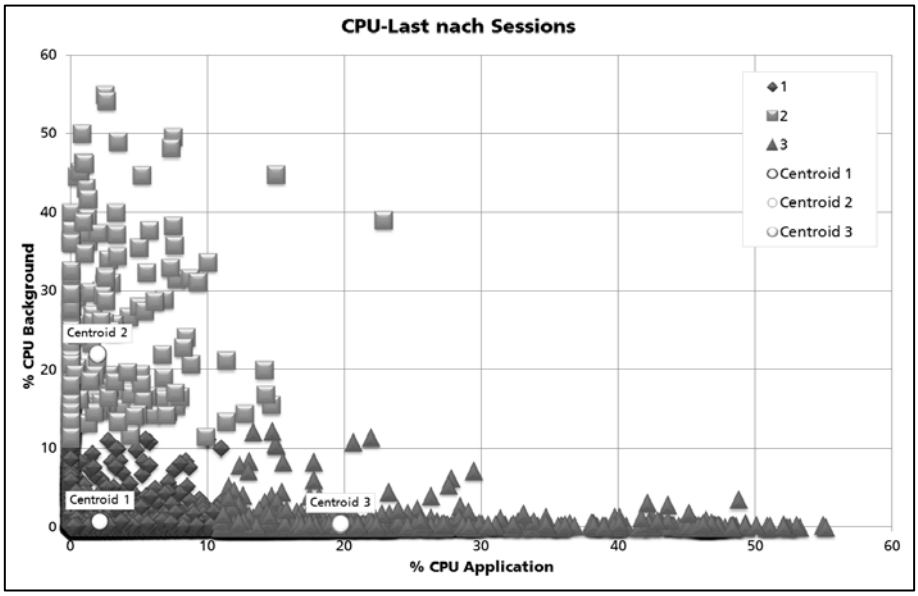
Timestamp	Session ID	App (%)	System (%)	Total (%)
(…)	(…)	(…)	(…)	(…)
23.06.2014 10:00	104361	0,19	0,01	0,20
23.06.2014 10:00	104362	8,05	0,06	8,10
23.06.2014 10:00	104368	4,34	0,05	4,38
23.06.2014 10:00	104370	1,10	0,04	1,15
23.06.2014 10:00	104372	3,20	0,03	3,23
(…)	(…)	(…)	(…)	(…)

Quelle: Eigene Darstellung

Die so strukturierten Daten werden dann einer Clusteranalyse mittels k-Means-Algorithmus (Kapitel 11.3) unterzogen[106], wobei gemäß den eingangs vorgestellten Klassen der *Light*, *Medium* und *Power User* drei Cluster angenommen werden ($k = 3$). Das Diagramm (Abbildung 6-9) stellt die Ergebnisse grafisch dar, wobei die Last durch Anwendungen auf der X- und die Last durch Systemprozesse auf der Y-Achse aufgetragen sind.

[106] Im Kontext dieses Dokuments kommt eine frei verfügbare Implementierung des k-Means-Algorithmus des Entwicklers Sheldon Neilson zum Einsatz. Die Implementierung basiert auf einem in einer Microsoft Excel-Arbeitsmappe integrierten Visual BASIC for Applications (VBA) Makro, siehe http://www.neilson.co.za/k-means-cluster-analysis-in-microsoft-excel/ (abgerufen am 01.02.2016).

Abbildung 6-9: CPU-Last pro Zeiteinheit und Session (Clusteranalyse)



Quelle: Eigene Darstellung

Es zeigt sich dabei für die exemplarische Datenbasis, dass mit 92 Prozent die große Mehrheit der Datensätze geringe Ressourcenanforderungen stellt, während eine kleinere Gruppe von sieben Prozent höhere Anforderungen auf Seiten der Anwendungen verzeichnet. Nur ca. ein Prozent der Datensätze weist höhere Last im Bereich des Systems auf. In letzterem Fall handelt es sich um sporadisch auftretende, kurzfristige Lasten durch Systemdienste und Hintergrundprozesse.

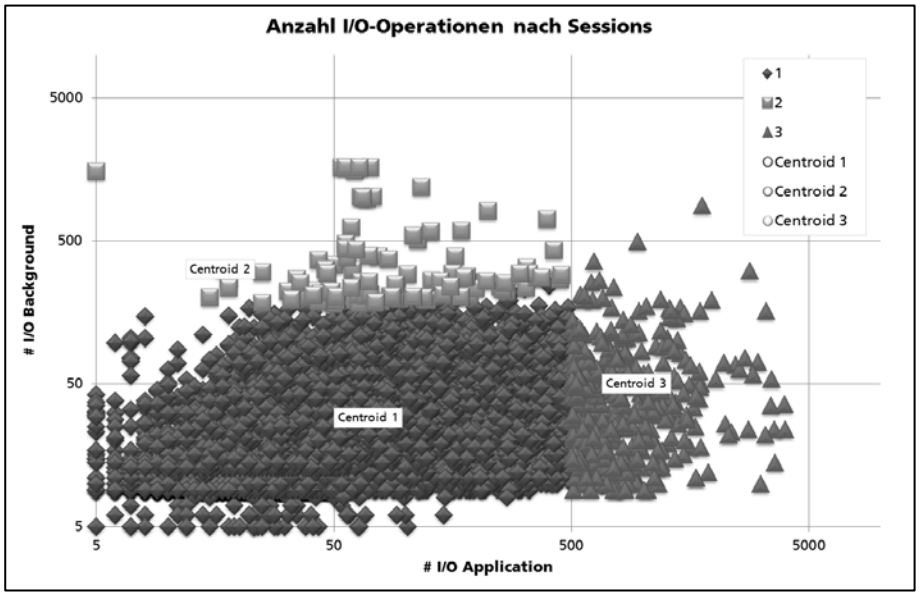
Tabelle 6-12: CPU-Last pro Zeiteinheit und Session (Clusteranalyse)

k-Means	App (%)	System (%)	Sessions (#)	Anteil (%)
Centroid 1	2,12	0,73	17537	92
Centroid 2	1,96	21,95	268	1
Centroid 3	19,69	0,41	1334	7
		Gesamt	**19139**	**100,00**

Quelle: Eigene Darstellung

Wird dieselbe Auswertung bezogen auf die Anzahl der I/O-Operationen pro Sekunde durchgeführt, so zeigt sich wiederum, dass die Mehrheit der Bedienwünsche auf einen Cluster entfällt (Abbildung 6-10).

Abbildung 6-10: IOPS pro Zeiteinheit und Session (Clusteranalyse)



Quelle: Eigene Darstellung

Innerhalb der exemplarischen Datenbasis weisen 88 Prozent der Datensätze eine niedrige Anzahl an Operationen auf, wobei die mit direktem Bezug zu den Anwendungen überwiegen (Tabelle 6-13). Zehn Prozent entfallen auf Systemprozesse und lediglich zwei Prozent der Datensätze zeigen deutlich höhere Last im Bereich der Anwendungen.

Tabelle 6-13: IOPS pro Zeiteinheit und Session (Clusteranalyse)

k-Means	App (#)	System (#)	Sessions (#)	Anteil (%)
Centroid 1	70	29	16863	88
Centroid 2	17	316	1860	10
Centroid 3	936	50	416	2
		Gesamt	**19139**	**100**

Quelle: Eigene Darstellung

Dies legt eine Modellierung der Benutzer in 1 bis 2 Klassen nahe und es wird der Ansatz gewählt, im Modell weiterhin nur eine Klasse zu Grunde zu legen, um die Komplexität zu reduzieren. Dies begründet sich darin, dass zur Unterscheidung der verschiedenen Typen von Client-Computern sowie der geografischen Standorte bereits sieben Lastketten erforderlich sind. Wären darüber hinaus auch mehrere Klassen von Anwendern zu unterscheiden, würde dies die Zahl der Lastketten erhöhen – bei einer Anzahl verschiedener Client-Anwendungsfälle n sowie einer Anzahl von Benutzerklassen k entsprechend auf $n \times k$ Lastketten.

Dem damit verbundenen Aufwand steht jedoch im betrachteten Szenario kein signifikanter Erkenntnisgewinn gegenüber, da sich die Sitzungen bezüglich ihrer Anforderungen an die Netzwerkbandbreite dem Prinzips der entfernten Präsentation folgend kaum unterscheiden

und die durch die zentral ausgeführten Anwendungen hervorgerufenen Lasten unabhängig von der Herkunft des Clients innerhalb des Rechenzentrums auftreten.

Die weitere Klassifizierung der Anwendungen erfolgt daher stattdessen anhand der Leistungsanforderungen der innerhalb der Sitzungen ausgeführten Prozesse, deren Gruppierung in Anwendungen geringer, mittlerer und hoher Komplexität mit entsprechender Erweiterung der Datenbank im Kapitel 12.5.4.3 dargelegt ist. Eine SQL-Abfrage der EdgeSight Datenbank (Tabelle 12-16) liefert die notwendigen Informationen, um für den eingeschränkten Beobachtungszeitraum die Anzahl ausgeführter Applikationen sowie deren Verteilung auf die Klassen zu ermitteln (Tabelle 6-14).

Tabelle 6-14: Ressourcenbedarf nach Komplexität der Anwendungen

Klasse	Anzahl	Anteil (%)
simple	72286	16
medium	189751	42
complex	58732	13
system	131018	29
Gesamt	**451787**	**100**

Quelle: Eigene Darstellung

Die so gewonnenen Informationen bezüglich der Verteilung der Anwender auf Gruppen von Client-Computern, Subnetze und Standorte sowie der Anwendungsprozesse dienten im Folgenden bei einem weiteren Durchlauf des Lebenszyklus der genaueren Parametrisierung der Lastketten im VITO-Modell (Tabelle 6-15).

Dabei wurden zunächst die ermittelten Werte bezüglich der geografischen Verteilung der Benutzer aus dem vorherigen Kapitel 6.5.3 berücksichtigt. Zum anderen wurden die Lastketten auf Seiten der Server in die verschiedenen Klassen von Applikationen und Systemprozessen differenziert. Diese Differenzierung erfolgte in der nachstehenden Tabelle allerdings exemplarisch nur für eine Client-Gruppe, nämlich wiederum für die größte Gruppe *TCs_HQ_Sessions* und die zugehörige Server-Last.

Tabelle 6-15: Lastketten des angepassten VITO-Modells

Nr.	Name	Population	Relation	Koeffizient
1	PCs_HQ_Sessions	18	1	1,00
2	TCs_HQ_Sessions	112	2	1,00
3	PCs_BranchA_Sessions	3	3	1,00
4	TCs_BranchA_Sessions	17	4	1,00
5	PCs_BranchB_Sessions	1	5	1,00
6	TCs_BranchB_Sessions	5	6	1,00
7	PCs_VPN_Sessions	16	7	1,00
8	PCs_HQ_Apps+System	18	1	1,00

Diese Relation koppelt die Lastkette 8 an Lastkette 1.

Nr.	Name	Population	Relation	Koeffizient
9	TCs_HQ_Apps_simple	112	2	0,16
10	TCs_HQ_Apps_medium	112	2	0,42
11	TCs_HQ_Apps_complex	112	2	0,13
12	TCs_HQ_System	112	2	0,29
13	PCs_BranchA_Apps+System	3	3	1,00
14	TCs_BranchA_Apps+System	17	4	1,00
15	PCs_BranchB_Apps+System	1	5	1,00
16	TCs_BranchB_Apps+System	5	6	1,00
17	PCs_VPN_Apps+System	16	7	1,00

Diese Relationen koppeln die Lastketten 9-12 an Lastkette 2.

Quelle: Eigene Darstellung

Die Server-Last wurde in vier neue Lastketten (*TCs_HQ_Apps_simple*, *TCs_HQ_Apps_medium*, *TCs_HQ_Apps_medium* und *TCs_HQ_System*) aufgeteilt, deren Durchsätze über die Relation an *TCs_HQ_Sessions* gekoppelt sind. Der zuvor ermittelte Ressourcenbedarf nach Komplexität der Anwendungen wird dabei über die Koeffizienten abgebildet.

Für die übrigen Client-Gruppen sind die Server-Lasten weiterhin in jeweils einer Lastkette zusammengefasst, da ansonsten zu diesen sechs Lastketten zusätzlich 18 weitere Ketten zu modellieren wären. Dies würde die Komplexität des Modells somit deutlich steigern und den Aufwand nur dann rechtfertigen, wenn pro Client-Gruppe unterschiedliches Verhalten auf Server-Seite abzubilden wäre. Da dies im gegenwärtigen Szenario noch nicht erforderlich war, wurden die Anwendungen nur für die größte Client-Gruppe im Detail betrachtet.

Tabelle 6-16: Lösung des angepassten VITO-Modells

Station	UtilUsr	UtilOhd	Util	Idle	Max.Util
RDSH01	48,49%	2,97%	51,45%	48,55%	60,00%
RDSH02	48,49%	2,97%	51,45%	48,55%	60,00%
RDSH03	48,49%	2,97%	51,45%	48,55%	60,00%
RDSH04	48,49%	2,97%	51,45%	48,55%	60,00%
RDSH05	48,49%	2,97%	51,45%	48,55%	60,00%
RDSH06	48,49%	2,97%	51,45%	48,55%	60,00%
RDSH07	48,49%	2,97%	51,45%	48,55%	60,00%
RDSH08	48,49%	2,97%	51,45%	48,55%	60,00%
RDSH09	48,49%	2,97%	51,45%	48,55%	60,00%
RDSH10	48,49%	2,97%	51,45%	48,55%	60,00%
(…)	(…)	(…)	(…)	(…)	(…)
RDSH15	48,49%	2,97%	51,45%	48,55%	60,00%
RDSH_Special01	19,14%	0,62%	19,76%	80,24%	60,00%
RDSH_Special02	19,14%	0,62%	19,76%	80,24%	60,00%

Zur Erläuterung der VITO-Ausgabe: Die Spalte „UtilUsr" (User) gibt die direkt durch den Benutzer erzeugte Prozessorlast an, die Spalte „UtilOhd" (Overhead) einen Zuschlag, der durch die Skalierung des Multiprozessorsystems hervorgerufen wird (vgl. Kapitel 11.2.2).

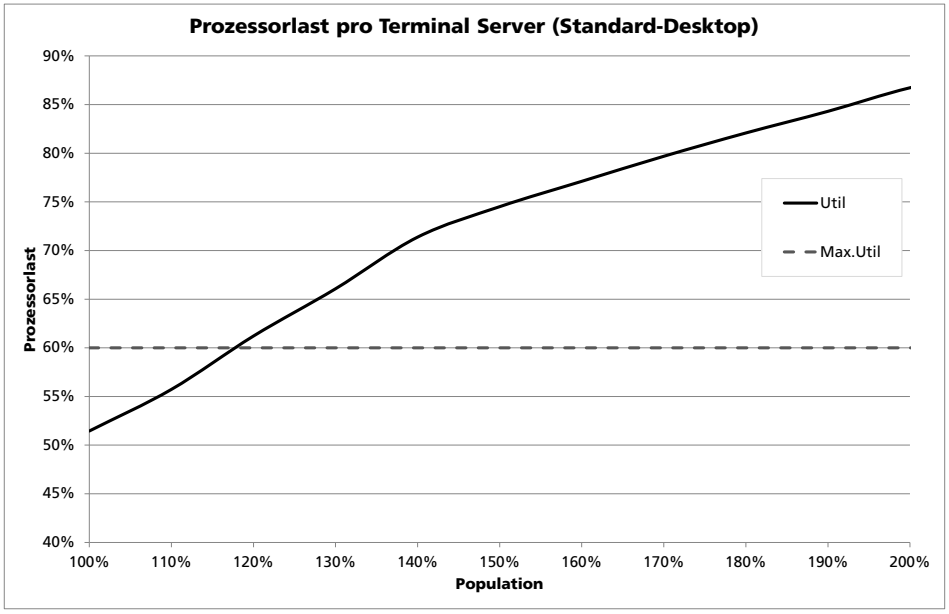
Die Spalte „Util" weist die kumulierte Prozessorlast aus, die Spalte „Idle" komplementär dazu den Leerlauf.

Der konfigurierbare Wert „Max. Util" markiert die Warnschwelle, bis zu der das System als benutzbar angenommen wird.

Quelle: Eigene Darstellung

Nach der Anpassung des Modells und seiner erneuten Parametrisierung mit den aus der ersten Betriebsphase gewonnenen Erkenntnissen konnte das Modell im nächsten Durchlauf des Lebenszyklus für die weitere Projektplanung verwendet werden. Die Lösung des überarbeiteten Modells zeigte entsprechend den veränderten Input-Daten höhere Prozessorlast von ca. 51 Prozent auf den Terminal Servern des Standard-Desktops (Tabelle 6-16), was kongruent war zu den bis dahin gewonnenen Erfahrungen aus dem praktischen Betrieb.

Abbildung 6-11: Variation der Population für das angepasste Modell



Quelle: Eigene Darstellung

Mit den angepassten Werten prognostizierte die Variation des Modells mit steigender Population das Erreichen der kritischen Lastgrenze bereits bei einer Zunahme der Sitzungen um 20 Prozent (Abbildung 6-11), was durch eine Auswertung im Hinblick auf die für die einzelnen Typen von Applikationen zu erwartenden Antwortzeiten noch unterstützt wurde.

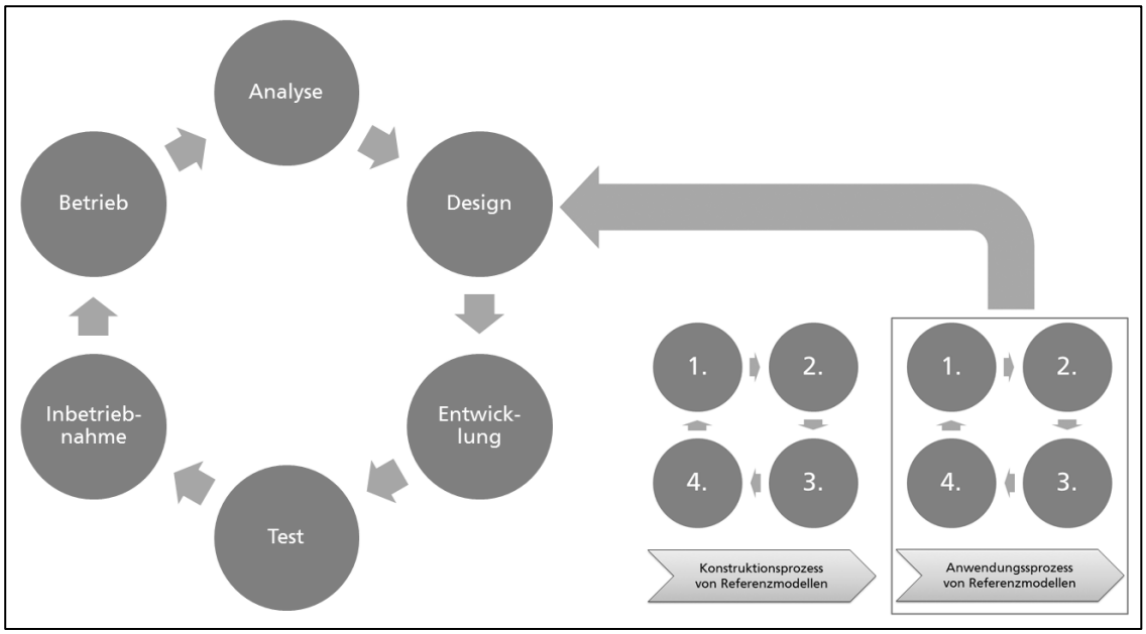
6.6 Schlussfolgerungen aus der Arbeit mit dem Modell

Es konnte schon mit Hilfe des ursprünglichen Modells, das mangels praktischer Erfahrungswerte auf Annahmen und Schätzungen basierte, eine SBC Infrastruktur geplant und implementiert werden, die entsprechend der Vorhersage tatsächlich in der Lage war, die vorgesehene Last zu tragen. Bei der Beobachtung dieser Implementierung zeigte sich dann allerdings, dass gegenüber den Gegebenheiten im praktischen Betrieb für die im Modell prognostizierte Gesamtlast etwas zu geringe Werte ermittelt worden waren, weshalb das Modell dann in einem erneuten Durchlauf des Lebenszyklus (Abbildung 6-12) entsprechend angepasst wurde.

Die in den Kapiteln und beschriebene Methodik, derzufolge aus dem abstrakten Referenzmodell ein konkreteres Informationsmodell, daraus wiederum ein noch konkreteres

Simulationsmodell in VITO und schließlich eine praktische Implementierung abgeleitet wird, wurde also wiederholt angewendet. Somit konnten genauere Grenzen für die Leistungsfähigkeit der SBC Infrastruktur berechnet werden, welche Antworten auf die eingangs gestellten Fragestellungen zulassen.

Abbildung 6-12: Referenzmodell – Integration in den Lebenszyklus



Quelle: Eigene Darstellung

Es zeigte sich, dass die Umgebung den Ausfall eines oder mehrer virtueller Terminal Server und auch den Ausfall eines Virtualisierungshosts kurzfristig toleriert. Es ist dann allerdings davon auszugehen, dass das vom Benutzer wahrgenommene Antwortverhalten leidet, so dass die Umgebung in einem solchen Zustand nicht dauerhaft betrieben werden kann. Entsprechend ist mindestens ein weiterer Virtualisierungshost mit weiteren virtuellen Terminal Servern vorzusehen, um die zum Ausgleich einer Störung nötige Redundanz zu gewährleisten. Eben dies gilt auch im Falle einer Zunahme der Gesamtlast um 10 bis 20 Prozent, etwa durch ein Wachstum der Belegschaft und Anzahl an Endgeräten über alle Standorte hinweg oder die Integration eines neuen Standorts.

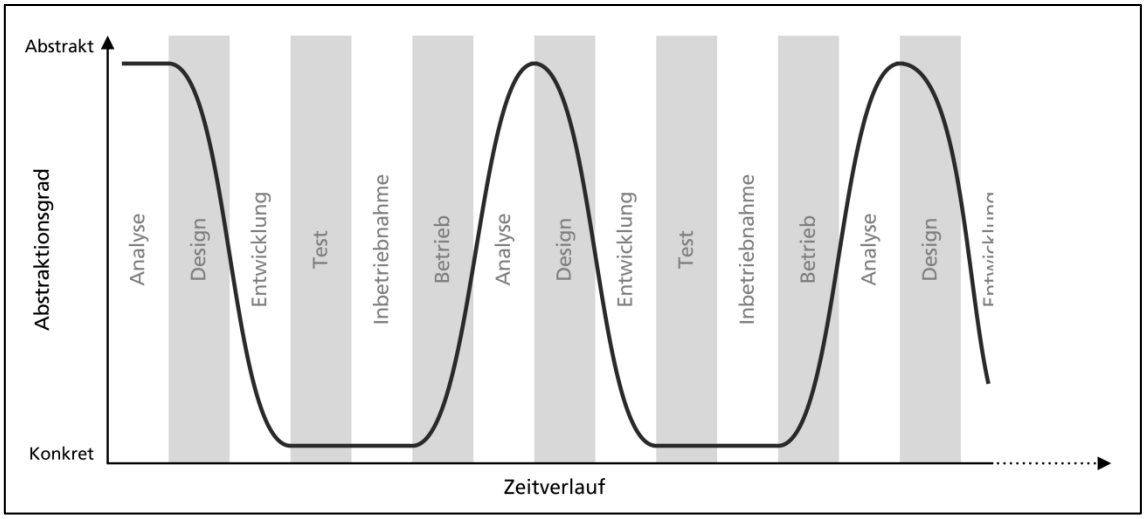
Beim wiederholten Durchlaufen des Lebenszyklus wechselt fortwährend der Abstraktionsgrad der Betrachtung, beginnend mit einer abstrakten Beschreibung der Anforderungen an die zu modellierende Umgebung in der Analysephase[107] bis hin zur konkreten Implementierung einer

[107] Zur Abgrenzung der Begriffe sei explizit darauf hingewiesen, dass die „Analyse" im Kontext des Lebenszyklus die Spezifikation der funktionalen und nicht-funktionalen Anforderungen aus Kundensicht bezeichnet, welche die SBC Infrastruktur erfüllen soll (vgl. Kapitel 4.3.1). Dieser Begriff ist somit zu unterscheiden von der „Analytik" im Sinne der Untersuchung eines mathematischen Modells mittels analytischer Methoden, wie sie bei der weiteren Verwendung von Informationsmodellen zum Einsatz kommen (vgl. Kapitel 5.4.4.2).

SBC Infrastruktur in der Phase des praktischen Betriebs. Im weiteren Verlauf einer daran anschließenden neuen Analysephase – mit ggf. veränderten Anforderungen seitens der Endanwender und sich wandelnden technischen Randbedingungen – wechselt der Prozess wiederum in eine abstraktere Perspektive, um dann von Neuem zu beginnen (Abbildung 6-13).

Der Einsatz von Ansätzen der Referenzmodellierung hat sich damit im Hinblick auf die Methodik als nützliches Werkzeug bei der Konzeption und Weiterentwicklung von SBC Infrastrukturen erwiesen. Es zeigte sich aber auch, dass ein Informationsmodell bzw. das daraus abgeleitete VITO-Modell nur so gut sein kann wie die Parameter, auf denen es basiert. Hier stand bei der Diskussion des Modells insbesondere die Prozessorlast im Vordergrund, da diese zunächst auf aus der Praxis gewonnenen Erfahrungswerten und erst im weiteren Verlauf auf fundierten Messungen basierte.

Abbildung 6-13: Abstraktionsgrad der Modellierung im Zeitverlauf



Quelle: Eigene Darstellung

In den Bereichen der Netzwerkkomponenten und insbesondere des Dateiservers konnte im vorliegenden Beispiel keine genauere Parametrisierung vorgenommen werden werden, da genaue Messwerte an jeder einzelnen Systemkomponente, wie z. B. Terminal Server, Provisioning Server, XenServer, Netzwerkkomponenten oder Dateiserver, auf Grund von vielfältigen Interdependenzen sowie mangels entsprechender Messpunkte und -technik nicht ermittelt werden konnten (Kapitel 12.1). Dies wird Gegenstand von Folgearbeiten sein (Kapitel 7.2) und betrifft nur die weitere Verwendung des speziellen Informationsmodells in diesem vorliegenden Beispielszenario, nicht den generellen methodischen Ansatz der Referenzmodellierung.

Die grundlegende Methodik der Referenzmodellierung wurde von den bei der Konzeption und Implementierung des Beispielszenarios involvierten IT-Systemtechnikern und Administratoren grundsätzlich positiv aufgenommen. Insbesondere die Verwendung der FMC Blockdiagramme beim Schritt vom allgemeinen Referenzmodell hin zum konkreten Informationsmodell hat sich

als äußerst nützlich erwiesen, da die Sprachelemente der FMC eine einheitliche Syntax bereitstellen, mit der Systemlandschaften beschrieben werden können. Das *Referenzmodell für das Kapazitätsmanagement von Server Based Computing Infrastrukturen* wird entsprechend auch bei der zukünftigten Weiterentwicklung dieses und weiterer Einsatzszenarien Verwendung finden.

7 Zusammenfassung und Ausblick

7.1 Zusammenfassung der Arbeit und ihrer Ergebnisse

Im Hinblick auf den rasant fortschreitenden Wandel der Arbeitsweisen und Nutzung von IT-Ressourcen gewinnen in Unternehmen branchenübergreifend Technologien zur effizienten und effektiven Bereitstellung von Applikationen und Desktop-Umgebungen zunehmend an Bedeutung. Endanwender wollen IT-Ressourcen nicht mehr nur stationär und an fixen Büroarbeitsplätzen in den Liegenschaften ihres Unternehmens nutzen, sondern auch mobil – etwa von zu Hause, unterwegs sowie auch an national und international verteilten Standorten ihres Unternehmens oder aus Netzen von Kunden oder Projektpartnern heraus. Vielfach *wollen* sie dies nicht nur, ja *müssen* es vielmehr, um ihre Aufgaben zu erfüllen.

Die Architektur von Systemlandschaften für den Betrieb von Desktops und Anwendungen hat sich in den zurückliegenden Jahrzehnten entsprechend signifikant gewandelt. Neben einer zunehmenden Verbreitung von Notebooks, welche die Verkaufszahlen klassischer Desktop-Computer rapide sinken lässt und das Arbeiten nach dem herkömmlichen Prinzip der verteilten Datenhaltung und -verarbeitung unterstützt, ist ein weiterer gegenläufiger Trend zu beobachten. So erlauben Unternehmen ihren Endanwendern zunehmend auch die Nutzung privater Endgeräte („Bring Your Own Device (BYOD)“). Marktbeobachter erwarten, dass dieser Trend sich in den kommenden Jahren nicht nur beschleunigen, sondern auch einen Paradigmenwechsel einleiten wird, so dass viele Unternehmen die Nutzung privater Geräte nicht nur *erlauben*, sondern sogar *verlangen*.

Im Zuge dieser Entwicklung sinkt die Bedeutung von Microsoft Windows als monolithischer Arbeitsumgebung, die einen kompletten Desktop mitsamt allen Anwendungen umfasst. Vielmehr wächst die Vielfalt der Endgeräte und Betriebssystemplattformen auf Seiten der Endanwender, während jedoch gleichzeitig einzelne Windows-Applikationen nicht aussterben werden, da sie vielfach unternehmenskritische Funktionen unterstützen. Diese Anwendungen ließen sich nicht oder nur mit enormem Aufwand auf andere Plattformen portieren. Vor diesem Hintergrund ist das Server Based Computing (SBC) ein wichtiger Baustein, um die Nutzung des existierenden Altbestands an Applikationen nach dem Prinzip der entfernten Präsentation auch von abweichenden Plattformen und Endgerätetypen aus zu ermöglichen.

Die Leistungsfähigkeit entsprechender IT-Infrastrukturen in den Rechenzentren hängt jedoch maßgeblich von der Kapazität der zugehörigen Server und weiteren Komponenten im Rechenzentrum ab und es folgt sowohl unter ökologischen als auch unter ökonomischen Erwägungen die Anforderung, vorhandene Hardware- und Software-Plattformen möglichst passend zum heutigen wie auch zukünftigen Bedarf zu dimensionieren und die Systeme optimal auszulasten.

7.1.1 Zielsetzung

In diesem Kontext ist der Gegenstand der hiermit vorliegenden Dissertation die Entwicklung und Erprobung eines *Referenzmodells für Server Based Computing Infrastrukturen und dessen Anwendung für das Kapazitätsmanagement*. Es leistet einen wertvollen Beitrag, vorhandene wie auch neu zu konzipierende Systeme zu planen und in einem iterativen Prozess weiterzuentwickeln. Das primäre Ziel ist dabei, einen Ansatz zu erarbeiten, der Administratoren und IT-Systemtechniker im Bereich des Server Based Computings bei der Analyse, Bewertung und Weiterentwicklung ihrer Systemlandschaften unterstützt und ihnen hierzu eine gemeinsame Sprache als Basis bereitstellt. Das Modell soll zudem eine erste Prognose bezüglich der möglichen Kapazität der Infrastrukturen ermöglichen, ohne dass vorab der tatsächliche Aufbau eines physischen Prototyps des gewünschten Zielsystems und die Simulation von Arbeitslasten darauf notwendig wären.

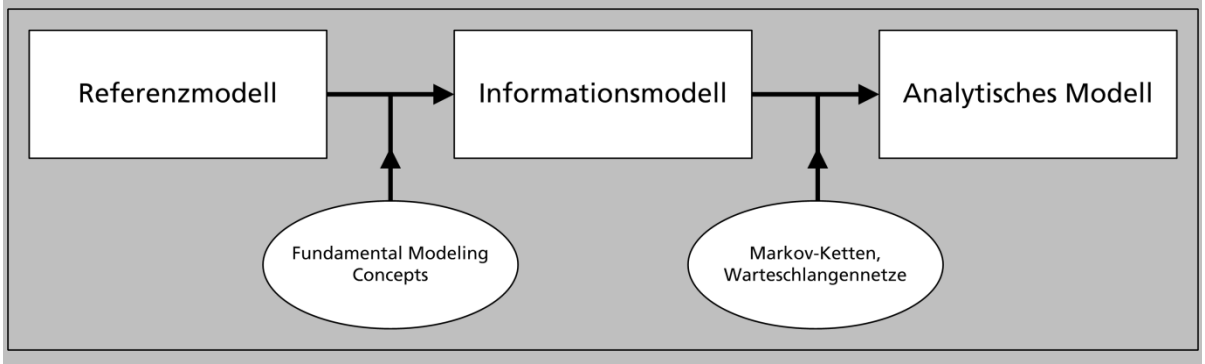
Zur Erreichung dieses Ziels wurde zunächst der Stand der Technik untersucht und es wurden zentrale technische Begrifflichkeiten geklärt sowie der Betrachtungsraum abgegrenzt (Kapitel 2 und 3). Im Hinblick auf das methodische Herangehen wurden weiterhin allgemeine Aspekte und Ziele des Kapazitätsmanagements herausgearbeitet und im Hinblick auf die Planung von SBC Infrastrukturen konkretisiert (Kapitel 4).

Anschließend erfolgte eine umfassende Literaturrecherche zu Grundlagen und Begrifflichkeiten der Referenzmodellierung sowie zu bereits existierenden Referenzmodellen verschiedener Forschungsschwerpunkte und Branchen.

7.1.2 Das Referenzmodell

Daraufhin werden die Lösungsbausteine des eigenen Ansatzes und somit der Kern *eines Referenzmodells für das Kapazitätsmanagement von Server Based Computing Infrastrukturen* mitsamt einer Methodik zu dessen Verwendung entwickelt, wobei letztere die logische Abfolge der nötigen Arbeitsschritte beschreibt (Kapitel 5). **Modell und Methodik** stellen ein Werkzeug bereit, das in dieser Form im Bereich des Server Based Computings zuvor nicht existierte, und bilden so den maßgeblichen wissenschaftlichen Beitrag der Arbeit.

Abbildung 7-1: Methodik zur Anwendung des Referenzmodells



Quelle: Eigene Darstellung

Die Methodik der Referenzmodellierung umfasst zwei Schritte: Zunächst wird aus dem Referenzmodell ein konkreteres Informationsmodell abgeleitet, welches eine qualitative Darstellung und Beschreibung der zu schaffenden SBC Infrastruktur bietet. Im zweiten Schritt wird dieses Modell in ein analytisches Modell überführt, welches quantitative Aussagen über Kapazität und Leistungsfähigkeit der SBC Infrastruktur erlaubt (Abbildung 7-1).

Es wird dazu zunächst die aus fünf Schichten bestehende Gesamtsicht einer Server Based Computing Infrastruktur entworfen. Aus diesem Referenzmodell werden nach einem methodischen Vorgehen konkretere Informationsmodelle abgeleitet und dann in der Sprache der Fundamental Modeling Concepts (FMC) notiert.

Die Sprachelemente der FMC Blockdiagramme bilden als Werkzeug zur Ableitung konkreter Informationsmodelle aus dem Referenzmodell die Basis, um entsprechende Systemlandschaften darzustellen – und das unabhängig davon, welcher Weg bei einer konkreten Konzeption und Implementierung gewählt wird. Neben der Aufbaustruktur einer SBC-Implementierung mit Terminal Servern oder virtuellen Desktops lassen sich auch dynamische Strukturen zu deren variablen Erzeugung zur Laufzeit notieren. So schafft die Referenzmodellierung mit ihren methodischen Werkzeugen für IT-Systemtechniker und Administratoren eine gemeinsame Grundlage zur Verständigung. Ein solches Modell kann anschließend im Rahmen einer Simulation oder einer analytischen Herangehensweise dazu verwendet werden, bereits bei der Konzeption verschiedene Handlungsalternativen zu untersuchen und bezüglich der möglichen Kapazität der Ressourcen zu bewerten.

Durch die Einführung der FMC und ihrer Sprachelemente in den Bereich des Server Based Computings wird das eingangs gesetzte Ziel adressiert, für Administratoren und IT-Systemtechnikern eine gemeinsame verbindlichere Basis für die Analyse, Bewertung und Weiterentwicklung ihrer Systemlandschaften zu schaffen. Eine solche verbindliche Grundlage mit einer definierten Syntax war zuvor im Bereich des Server Based Computings noch nicht existent.

Das Referenzmodell und seine Methodik werden schließlich anhand eines exemplarischen Szenarios mit verschiedenen Gruppen von Anwendern und Arbeitsplatzgeräten auf der Client-Seite und mehreren Profilen von Anwendungen auf der Server-Seite erprobt und validiert (Kapitel 6).

Die theoretische Grundlage für das analytische Modell bilden dabei Markov-Ketten und Warteschlangennetze mit den entsprechenden Algorithmen zu ihrer Lösung, durch die eine quantitative Bewertung der SBC Infrastruktur im Beispielszenario möglich wird.

Hiermit wird deutlich, dass die modellbasierte Herangehensweise bei wiederholter Anwendung im Lebenszyklus einer SBC Infrastruktur einen wertvollen Beitrag zum Kapazitätsmanagement leisten kann. Dabei ist kein tatsächlicher Aufbau einer neuen IT-Infrastruktur mittels Bau eines physischen Prototypen und der Simulation von Arbeitslasten notwendig. Damit wurden die eingangs gestellten Forschungsfragen beantwortet und das zu Beginn gesetzte Ziel der Arbeit erreicht.

7.2 Ausblick

Mit dem im Rahmen dieser Dissertation entwickelten *Referenzmodell für das Kapazitätsmanagement von Server Based Computing Infrastrukturen* liegt ein im praktischen Einsatz erprobtes Werkzeug zur methodischen Unterstützung von IT-Systemtechnikern und Administratoren bei der Planung und Weiterentwicklung entsprechender Systemlandschaften vor. Das Referenzmodell beschreibt eine aus fünf Schichten aufgebaute Struktur, welche die Grundlage bildet, um in einem „Top-Down"-Ansatz mittels eines methodischen Vorgehens konkretere Informationsmodelle daraus abzuleiten.

Im Hinblick auf die bei der Erprobung und Validierung gemachten Erfahrungen und die dabei gewonnenen Erkenntnisse können für die zukünftige Verwendung des Referenzmodells und seiner Methodik mögliche Ansatzpunkte für folgende Arbeiten formuliert werden. So wurden die in Abbildung 7-1 dargestellten Schritte zur Anwendung der Methodik im Rahmen dieser Arbeit manuell ausgeführt. Der Anwendungsweg stellte sich somit folgendermaßen dar – vom Referenzmodell zu einem konkreten in FMC notierten Informationsmodell, von dort weiter zu einem in VITO modellierten Warteschlangennetzwerk und schließlich vom theoretischen Modell zu einer praktischen Implementierung. Für eine zukünftige Vereinfachung dieses Prozesses ist hinsichtlich der verwendeten semi-formalen Sprache der FMC die Frage zu beantworten, wie modellierte Sachverhalte syntaktisch und semantisch auf ihre Korrektheit hin überprüft werden können.

Eine mögliche Automatisierung dieses Prozesses wird Gegenstand zukünftiger Forschungsfragen sein und soll in ein Software-Werkzeug münden, welches es in Form eines integrierten Editors erlaubt, FMC-Diagramme zu entwerfen, zu parametrisieren, automatisch in ein analytisches Modell zu überführen und dieses schließlich auch zu lösen. Hier kann allerdings nur eine bessere Unterstützung durch Werkzeuge das Ziel sein. Eine vollständige Automatisierung des Prozesses wird dagegen kaum möglich sein, da beim Entwurf immer das spezifische Fachwissen eines Administrators oder IT-Systemtechnikers über die realweltlichen Gegebenheiten und Randbedinungen des Einsatzes von SBC erforderlich sein wird.

Weitere zukünftige Fragen betreffen vornehmlich die Anwendung der Methodik im zweiten bzw. in wiederholten Durchläufen des Lebenszyklus (Kapitel 4.3) – also Szenarien, in denen eine SBC Infrastruktur bereits existiert und in den produktiven Betrieb überführt wurde. Hier musste bei der konkreten Verwendung insbesondere im Bereich des Storage teilweise mit Annahmen gearbeitet werden, da genaue Messwerte an jeder einzelnen Systemkomponente, wie z. B. Terminal Server, Provisioning Server, XenServer, Netzwerkkomponenten oder Dateiserver, auf Grund von vielfältigen Interdependenzen sowie mangels entsprechender Messpunkte und -technik nicht ermittelt werden konnten (Kapitel 12.1).

Weiterhin erfolgte die Erprobung unter Verwendung bestimmter Ausprägungen und Versionen von SBC-Softwareprodukten (Microsoft Windows Server 2008 R2, Citrix XenApp 6.5, Citrix Provisioning Services 7.x, Citrix XenServer 6.x). Ebenso wie die Architektur von XenApp grundlegend weiterentwickelt wurde (Kapitel 2.3.2), wurde in diesem Zug auch das Monitoring komplett neu implementiert. Wenn das hier entwickelte Referenzmodell auf

neuere Versionen der Software oder andere Produkte bzw. Hersteller sowie auf das Konzept der Desktop-Virtualisierung erweitert werden soll, sind entsprechend alternative Wege zur Gewinnung der nötigen Messdaten zu wählen, um das Modell zu parametrisieren (Kapitel 12.7).

Eine zusätzliche Herausforderung besteht in der Frage, wie der Einsatz von Methoden der Referenzmodellierung quantitativ und qualitativ zu bewerten ist bzw. welche bewerteten Größen hierzu herangezogen werden können (vgl. Kapitel 5.5). Im Hinblick auf SBC Infrastrukturen ist eine umfangreichere Datenbasis aus dem Einsatz des Referenzmodells in vielen verschiedenen Einsatzszenarien noch nicht vorhanden und es wird Gegenstand weiterer Forschung sein, die im Rahmen dieser Arbeit entwickelte Methodik in zukünftigen Projekten einzusetzen und Vergleiche zwischen diesen Projekten anzustellen. Entsprechende Erfahrungswerte werden die Basis bilden, um zu einer quantitativen Bewertung der Vorteile zu gelangen, welche sich auch unter betriebswirtschaftlichen Aspekten durch eine Optimierung und Verkürzung der Planungsprozesse von SBC Infrastrukturen realisieren lassen.

Die zuvor aufgeführten Fragen werden Gegenstand von Folgeaktivitäten zu diesem Thema sein, die ihren Beitrag dazu leisten, dass das *Referenzmodell für das Kapazitätsmanagement von Server Based Computing Infrastrukturen* auch zukünftig beim Einsatz neuerer Technologien und Produkte im Bereich des SBC einen nützlichen Beitrag zur Planung entsprechender Systemlandschaft leistet.

8 Abkürzungsverzeichnis

Abkürzung	Bedeutung
3D	Drei Dimensionen/dreidimensional
AAC	Advanced Audio Coding
ABI	Application Binary Interface
ACM	Association for Computing Machinery
AD	Active Directory
ADSL	Asymmetric Digital Subscriber Line
AES	Advanced Encryption Standard
AG	Access Gateway
AIE	Application Isolation Environment
AIX	Advanced Interactive Executive
AMD	Advanced Micro Devices, Inc.
AMD-V	AMD Virtualization
Anm. d. Verf.	Anmerkung des Verfassers
API	Application Programming Interface
App-V	(Microsoft) Application Virtualization
ASIC	Application-Specific Integrated Circuit
ASP	Active Server Pages
ASPLOS	Architectural support for programming languages and operating systems
ATM	Asynchronous Transfer Mode
AVI	Audio Video Interleave
BASIC	Beginner's All-Purpose Symbolic Instruction Code
BIOS	Basic Input/Output System
BITKOM	Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e.V.
BMBF	Bundesministerium für Bildung und Forschung
BOOTP	Bootstrap Protocol
BSD	Berkeley Software Distribution
BSI	Bundesamt für Sicherheit in der Informationstechnik
BT	Binäre Translation (Binary Translation)
BYOD	Bring Your Own Device
bzw.	beziehungsweise
CA	Certification Authority
CAD	Computer Aided Design
CDE	Common Desktop Environment
CDN	Citrix Developer Network

Abkürzung	Bedeutung
CGP	Common Gateway Protocol
chroot	change root
CIFS	Common Internet File System
CMS	Conversational Monitor System
Codec	Compression/Decompression
CP	Control Program
CPU	Central Processing Unit
CRM	Customer Relationship Management
CSG	Citrix Secure Gateway
CSIM	Complete Software Interpreter Machine
CSP	Citrix Service Provider (Program)
CSV	Comma Separated Values
CTSS	Compatible Time-sharing System
d. h.	das heißt
DaaS	Desktop as a Service
DAS	Direct Attached Storage
DB	Datenbank
DDC	Desktop Delivery Controller
DEC	Digital Equipment Corporation
DH	Diffie Hellman
DHCP	Dynamic Host Configuration Protocol
DLL	Dynamic Link Library
DMA	Direct Memory Access
DMZ	Demilitarisierte Zone
DNS	Domain Name System
Dom0	Domain-0
DomU	Unprivileged Domain
DOS	Disk Operating System
DP	Desktop Publishing
DSL	Digital Subscriber Line
EC2	(Amazon) Elastic Compute Cloud
ECBS	Engineering of Computer-Based Systems
EPK	Ereignisgesteuerte Prozesskette
ER	Entity Relationship (Diagramm)
ERM	Entity Relationship Model
FCFS	First-Come-First-Serve
FIFO	First-In-First-Out
FK	Foreign Key

Abkürzung	Bedeutung
FMA	Flexcast Management Architechture
FMC	Fundamental Modeling Concepts
FQDN	Full Qualified Domain Name
FTP	File Transfer Protocol
FVM	Feather-Weight Virtual Machine
GB(yte)	Gigabyte
GB(yte)/s	Gigabyte pro Sekunde
Gbit	Gigabit
Gbit/s	Gigabit pro Sekunde
ggf.	gegebenenfalls
GHz	Gigahertz
GNOME	GNU Network Object Model Environment
GNU	Gnu's Not Unix
GPO	Group Policy Object
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HA	Hochverfügbarkeit (High Availability)
HBA	Host Bus Adapter
HD	High Definition
HDD	Festplatte (Hard Disk Drive)
HDX	High Definition User Experience
HP	Hewlett-Packard
HSPA	High Speed Packet Access
HTML	Hyper Text Markup Language
HTML5	Fünfte Fassung der Hyper Text Markup Language
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol over SSL
i. d. R.	in der Regel
I/O	Input/Output
IA-32	Intel Architecture 32-Bit
IA-64	Intel Architecture 64-Bit
IaaS	Infrastructure as a Service
IBM	International Business Machines Corporation
ICA	Independent Computing Architecture
ICT	Information and Communication Technology
ID	Identifier
IDC	International Data Corporation
IEEE	Institute of Electrical and Electronics Engineers

Abkürzung	Bedeutung
IIS	(Microsoft) Internet Information Server
IKE	Internet Key Exchange
IMA	Independent Management Architecture
IMAP	Internet Message Access Protocol
IOMMU	Input/Output Memory Management Unit
IOPS	Input/Output Operationen pro Sekunde
IP	Internet Protocol
IPC	Interprozesskommunikation (Inter Process Communication)
IPsec	Internet Protocol Security
IS	Infinite Server
ISAPI	Internet Server Application Programming Interface
iSCSI	SCSI over TCP/IP
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
ISYM	Information Systems & Management
IT	Informationstechnik
ITU	International Telecommunicatios Union
IVT	Intel Virtualization Technology
KB/Kbyte	Kilobyte
Kbit	Kilobit
KDE	K Desktop Environment
KI	Künstliche Intelligenz
LAN	Local Area Network
LCP	Large Customer Population (Algorithmus)
LDAP	Lightweight Directory Access Protocol
LUN	Logical Unit Number
LXC	Linux Containers
MAC	Media Access Control
MB(yte)	Megabyte
MB(yte)/s	Megabyte pro Sekunde
Mbit	Megabit
Mbit/s	Megabit pro Sekunde
MCS	Machine Creation Services
MDOP	Microsoft Desktop Optimization Pack
MEST	Mitteleuropäische Sommerzeit
MIB	Management Information Base
MIT	Massachusetts Institute of Technology
MMC	Microsoft Management Console

Abkürzung	Bedeutung
MP3	MPEG Layer-3 Audio
MP4	MPEG Layer-4 Audio
MPEG	Moving Picture Expert Group
MS-DOS	Microsoft Disk Operating System
MVA	Mean Value Analysis
NAS	Network Attached Storage
NC	Network Computer
NCRP	Network Computer Reference Profile
NetBIOS	Network Basic Input/Output System
NFS	Network File System
NT	(Windows) New Technology
OData	Open Data Protocol
OEM	Original Equipment Manufacturer
OGG	Ogg Vorbis
OID	Object Identifier
OR	Operational bzw. Operations Research
OS	Operating System
OSI	Open Systems Interconnection
OTP	One Time Password
OU	Organizational Unit
P-A-P	Paketfilter – Application-Level Gateway – Paketfilter
PaaS	Platform as a Service
PC	Personal Computer
PCI	Peripheral Component Interconnect
PCIe	Peripheral Component Interconnect Express
PCM	Power & Capacity Management
PCoIP	PC-over-IP
PDF	(Adobe) Portable Document Format
PDU	Protocol Data Unit
PE	Proportional Extimation (Algorithmus)
PIF	Physical Interface
PK	Primary Key
PKI	Public Key Infrastructure
PnP	Plug & Play
POTS	Plain Old Telephone System
PS	Processor Sharing
PSK	Pre-Shared Key
PV	Paravirtualisierung

Abkürzung	Bedeutung
PVC	Parallels Virtuozzo Containers
PVS	(Citrix) Provisioning Services
PXE	Preboot Execution Environment
QD	Queue Dependent
QEMU	Quick Emulator
QoS	Qualitiy of Service
RAM	Random Access Memory
RD-Gateway	(Microsoft) Remotedesktopgateway
RDP	Remote Desktop Protocol
RDS	Remotedesktopdienste (Remote Desktop Services)
RDSH	Remotedesktop-Sitzunghost (Remote Desktop Session Host)
ReCoBS	Remote-Controlled Browsers System
RFC	Request For Comment
RJ45	Registered Jack – 45
RRA	Round Robin Archive
RRD	Round Robin Database
RSA	Rivest, Shamir und Adleman
S3	(Amazon) Simple Storage Service
SaaS	Software as a Service
SAN	Storage Area Network
SBC	Server Based Computing
SCSI	Small Computer Systems Interface
SDL	Specification and Description Language
SIGOPS	Special Interest Group on Operating Systems
SIGPLAN	Special Interest Group on Programming Languages
SMB	Server Message Block
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SOSP	Symposium on operating systems principles
SP	Service Pack
SPARC	Scalable Processor Architecture
SQL	Structured Query Language
SRP	Software Restriction Policy
SSL	Secure Sockets Layer
SSYM	Security Symposium
TA/s	Transaktionen pro Sekunde
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol

Abkürzung	Bedeutung
TFTP	Trivial File Transfer Protocol
TLS	Transport Layer Security
TSE	(Windows NT) Terminal Server Edition
u. a.	unter anderem
UCS	Unified Computing System
UDP	User Datagram Protocol
UMSICHT	Fraunhofer-Institut für Umwelt-, Sicherheits- und Energietechnik
UML	Unified Modeling Language
UMTS	Universal Mobile Telecommunications System
UNIX	Uniplexed Information and Computing Service
URL	Uniform Resource Locator
USB	Universal Serial Bus
UTC	Coordinated Universal Time
UUID	Universally Unique Identifier
VBA	Visual BASIC for Applications
vCPU	Virtuelle Central Processing Unit
VDA	Virtual Desktop Agent
VDI	Virtual Desktop Infrastructure
vgl.	vergleiche
VHD	(Microsoft) Virtual Hard Disk
VIF	Virtual Interface
VM	Virtuelle Maschine
VMM	Virtual Machine Monitor, auch (Microsoft) Virtual Machine Manager
VPN	Virtual Private Network
VS-NfD	Verschlusssache – Nur für den Dienstgebrauch
VSI	(Login) Virtual Session Indexer
vSwitch	Virtueller Switch
VT-c	Intel Virtualization Technology For Connectivity
VT-d	Intel Virtualization Technology For Directed I/O
VT-i	Intel Virtualization Technology For Itanium Processors
VT-x	Intel Virtualization Technology For IA-32 And Intel 64 Processors
WAN	Wide Area Network
WAV	Windows Wave
WES	Windows Embedded Standard
WfW	Windows for Workgroups
WLAN	Wireless Local Area Network
WMA	Windows Media Audio
WMI	Windows Management Instrumentation

Abkürzung	Bedeutung
WMV	Windows Media Video
Xfce	XForms Common Environment
XML	Extensible Markup Language
z. B.	zum Beispiel

9 Literaturverzeichnis

Kurzzeichen	Quellenangabe
[Adams, 2006]	Adams, Keith; Agesen, Ole: „A Comparison of Software and Hardware Techniques for x86 Virtualization“, S. 2-13 in „Proceedings of the 12th international conference on Architectural support for programming languages and operating systems (ASPLOS-XII)“; Association for Computing Machinery, 2006
[Ahnert, 2007]	Ahnert, Sven: „Virtuelle Maschinen mit VMware und Microsoft – Für Entwicklung, Schulung, Test und Produktion“; Addison-Wesley, 2007
[AMD, 2011]	AMD (Hrsg.): „AMD I/O Virtualization Technology (IOMMU) Specification“; Advanced Micro Devices, 2011
[Anderson, 2010]	Anderson, Christa; Griffin, Kristin L.: „Windows Server 2008 R2 Remote Desktop Services – Resource Kit“; Microsoft Press, 2010
[Apfelbacher, 2004]	Apfelbacher, Rémy; Curth, André: „FMC and UML (Präsentation)“; FMC Research Staff, 2004
[Apfelbacher, 2005]	Apfelbacher, Rémy; Knöpfel, Andreas; Aschenbrenner, Peter; Preetz, Sebastian: „FMC Visualization Guidelines“; FMC Research Staff, 2005
[Apfelbacher, 2005a]	Apfelbacher, Rémy; Rozinat, Anne: „FMC Compositional Structures: Block Diagrams – Reference Sheet“; FMC Research Staff, 2005
[Bard, 1979]	Bard, Yonathan: „Some Extensions to Multiclass Queueing Network Analysis“, S. 51-62 in „Proceedings of the Third International Symposium on Modelling and Performance Evaluation of Computer Systems: Performance of Computer Systems“; North-Holland Publishing Co., 1979
[Barham, 2003]	Barham, Paul; Dragovic, Boris; Fraser, Keir; Hand, Steven; Harris, Tim; Ho, Alex; Neugebauer, Rolf; Pratt, Ian; Warfield, Andrew: „Xen and the Art of Virtualization“, S. 164-177 in „Proceedings of the nineteenth ACM symposium on operating systems principles (SOSP '03)“; Association for Computing Machinery, 2003
[Becker, 2003]	Becker, Jörg; Knackstedt, Ralf: „Konstruktion und Anwendung fachkonzeptioneller Referenzmodelle im Data Warehousing“, S. 415-433 in „Wirtschaftsinformatik 2003/Band II: Medien – Märkte – Mobilität“; Springer Verlag, 2003
[Becker, 2004]	Becker, Jörg; Schütte, Reinhard: „Handelsinformationssysteme: Domänenorientierte Einführung in die Wirtschaftsinformatik“; mi-Wirtschaftsbuch, 2004

Kurzzeichen	Quellenangabe
[BITKOM, 2009]	BITKOM Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (Hrsg.): „Cloud Computing – Evolution in der Technik, Revolution im Business"; BITKOM, 2009
[Brocke, 2003]	Brocke, Jan vom: „Referenzmodellierung – Gestaltung und Verteilung von Konstruktionsprozessen"; Logos Verlag, 2003
[BSI, 2006]	Bundesamt für Sicherheit in der Informationstechnik (Hrsg.): „Remote-Controlled Browsers System (ReCoBS) – Grundlagen und Anforderungen (Version 2.0, Stand: 23.06.2006)"; Bundesamt für Sicherheit in der Informationstechnik, 2006
[BSI, 2011]	Bundesamt für Sicherheit in der Informationstechnik (Hrsg.): „IT-Grundschutz-Kataloge – 12. Ergänzungslieferung (September 2011)"; Bundesamt für Sicherheit in der Informationstechnik, 2011
[Cheswick, 1994]	Cheswick, William R.; Bellovin, Steven M.: „Firewalls and Internet Security – Repelling the Wily Hacker"; Addison-Wesley, 1994
[Citrix, 2006]	Citrix Systems (Hrsg.): „Monitoring Server Performance with Citrix Presentation Server"; Citrix Systems, 2006
[Citrix, 2007]	Citrix Systems (Hrsg.): „Anatomy of an EdgeSight Report"; Citrix Systems, 2007
[Citrix, 2008]	Citrix Systems (Hrsg.): „Implementing Citrix XenServer Enterprise Edition 5.0 – CXS-200-1 Student Manual"; Citrix Systems, 2008
[Citrix, 2009]	Citrix Systems (Hrsg.): „Citrix EdgeSight Reporting Wiki (Beta) – PDF"; Citrix Systems, 2009
[Citrix, 2010]	Citrix Systems (Hrsg.): „Citrix XenClient Express – Proof of Concept Implementation Guide"; Citrix Systems, 2010
[Citrix, 2011]	Citrix Systems (Hrsg.): „Citrix Reference Architecture for Multi-Tenant Desktop as a Service – Using Citrix XenApp 6 to Enable the Delivery of Microsoft Windows-based Desktops and Software as a Service"; Citrix Systems, 2011
[Citrix, 2011a]	Citrix Systems (Hrsg.): „Citrix EdgeSight-Benutzerdokumentation (Citrix EdgeSight für Endpunkte 5.4/ Citrix EdgeSight für XenApp 5.4)"; Citrix Systems, 2011
[Creasy, 1981]	Creasy, R. J.: „The origin of the VM/370 time-sharing system", S. 483-490 in „IBM Journal of Research & Development – Vol. 25 – Nr. 5 – September 1981"; IBM, 1981
[Dannbacher, 2002]	Dannbacher, Andre; Kienle, Fabian: „Leitfaden für sichere Citrix-Systeme – Entwurf / Planung / Umsetzung"; Citrix Systems, 2002

Kurzzeichen	Quellenangabe
[Dapper, 1997]	Dapper, Thomas; Dietrich, Carsten; Klöppel, Bert; u.a.: „Windows NT 4.0 im professionellen Einsatz (2. Auflage, Band 1)"; Carl Hanser Verlag, 1997
[Delp, 2006]	Delp, Martin: „Ein Referenzmodell für die Herstellung von Fachmedienprodukten (IPA-IAO Forschung und Praxis – Nr. 433)"; Jost Jetter Verlag, 2006
[Echtle, 1990]	Echtle, Klaus: „Fehlertoleranzverfahren"; Springer Verlag, 1990
[Facchi, 1995]	Facchi, Christian: „Methodik zur formalen Spezifikation des ISO/OSI Schichtenmodells"; Fakultät für Informatik der Technischen Universität München, 1995
[Fettke, 2004]	Fettke, Peter; Loos, Peter: „Referenzmodellierungsforschung – Langfassung eines Aufsatzes (Paper 16)"; ISYM – Information Systems & Management, 2004
[Fettke, 2004a]	Fettke, Peter; Loos, Peter: „Systematische Erhebung von Referenzmodellen – Ergebnisse der Voruntersuchung (Paper 19)"; ISYM – Information Systems & Management, 2004
[Fettke, 2005]	Fettke, Peter; Loos, Peter: „Der Beitrag der Referenzmodellierung zum Business Engineering", S. 18-26 in „HMD – Praxis der Wirtschaftsinformatik 241"; dpunkt.verlag, 2005
[Forster, 2013]	Forster, Christoph: „Referenzmodell zur Gestaltung der Serviceorganisation in Unternehmen der Raumfahrtbranche zum Betrieb bemannter Raumfahrtsysteme (Stuttgarter Beiträge zur Produktionsforschung – Band 10)"; Fraunhofer Verlag, 2013
[Frank, 2000]	Frank, Ulrich: „Entwurf eines Referenzmodells für Handelsplattformen im Internet", in Tagungsband der Fachtagung „KnowTech2000 – Knowledge Engineering, Management, Consulting & Training, Leipzig"; BITKOM, 2000
[Gai, 2010]	Gai, Silvano; Salli, Tommi; Andersson, Roger: „Cisco Unified Computing System (UCS) – A Complete Reference Guide to the Cisco Data Center Virtualization Server Architecture"; Cisco Press, 2010
[Gartner, 2013]	Gartner, Inc. (Hrsg.): „Bring Your Own Device: The Facts and the Future"; Gartner, Inc., 2013
[Goldberg, 1972]	Goldberg, Robert P.: „Architectural Principles for Virtual Computer Systems – Ph.D. Thesis"; Harvard University, 1972
[Goldberg, 1973]	Goldberg, Robert P.: „Architecture of virtual machines", S. 74-112 in „Proceedings of the workshop on virtual computer systems"; Association for Computing Machinery, 1973
[Goldberg, 1974]	Goldberg, Robert P.: „A survey of virtual machine research", S. 34-45 in „Computer Magazine, Vol. 7 No. 6, June 1974"; IEEE Computer Society, 1974

Kurzzeichen	Quellenangabe
[Gröne, 2008]	Gröne, Bernhard; Knöpfel, Andreas; Kugel, Rudolf; Schmidt, Oliver: „The Apache Modeling Project"; Selbstverlag, 2008
[Hasan, 2013]	Hasan, Vaqar: „Instant EdgeSight for XenApp – Effective, practical instructions to monitor your Citrix XenApp servers using EdgeSight 5.4"; Packt Publishing, 2013
[Hiebel, 2008]	Hiebel, Markus; Knermann, Christian; Pflaum, Hartmut; Rettweiler, Manuela; Schröder, Andreas: „Ökologischer Vergleich der Klimarelevanz von PC und Thin Client Arbeitsplatzgeräten 2008"; Fraunhofer-Institut für Umwelt-, Sicherheits- und Energietechnik UMSICHT, 2008
[Hiebel, 2011]	Hiebel, Markus; Knermann, Christian; Maga, Daniel; Reinecke, André; Schröder, Andreas: „Thin Clients 2011 – Ökologische und ökonomische Aspekte virtueller Desktops"; Fraunhofer-Institut für Umwelt-, Sicherheits- und Energietechnik UMSICHT, 2011
[HP, 2010]	HP (Hrsg.): „Analyzing Citrix XenServer persistent performance metrics from Round Robin Database logs – Technical white paper"; Hewlett-Packard Development Company, 2010
[Hunt, 1992]	Hunt, Craig: „TCP/IP Network Administration"; O'Reilly & Associates, Inc., 1992
[IBM, 1997]	IBM (Hrsg.): „S/390 – IBM Network Station – Getting Started (November 1997)"; IBM, 1997
[IBM, 2005]	IBM (Hrsg.): „IBM Systems Virtualization Version 2 Release 1"; IBM, 2005
[Intel, 2010]	Intel Corporation (Hrsg.): „Intel Itanium Architecture – Volumes 1-4: Combined Reference Set, Revision 2.3"; Intel Corporation, 2010
[Intel, 2011]	Intel Corporation (Hrsg.): „Intel 64 and IA-32 Architectures – Software Developer's Manual – Combined Volumes: 1, 2A, 2B, 2C, 3A, 3B and 3C"; Intel Corporation, 2011
[Intel, 2011a]	Intel Corporation (Hrsg.): „Intel Virtualization Technology for Directed I/O – Architecture Specification"; Intel Corporation, 2011
[ISO, 1994]	International Organization for Standardization (Hrsg.): „ISO/IEC 7498-1:1994 – Information technology – Open Systems Interconnection – Basic Reference Model: The basic model"; International Organization for Standardization, 1994
[Kelkar, 2003]	Kelkar, Oliver: „Ein Referenzmodell für elektronische Geschäftstransaktionen im zwischenbetrieblichen Geschäftsverkehr (IPA-IAO Forschung und Praxis – Nr. 381)"; Jost Jetter Verlag, 2003

Kurzzeichen	Quellenangabe
[Keller, 2003]	Keller, Frank; Wendt, Siegfried: „FMC: An Approach Towards Architecture-Centric System Development", S. 173-182 in „Proceedings of the 10th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (ECBS)"; IEEE Computer Society, 2003
[Kempf, 2002]	Kempf, F.: „Referenzmodell zur integrierten Kommunikationsunterstützung von kooperierenden örtlich verteilten Akteuren (IPA-IAO Forschung und Praxis – Nr. 370)"; Jost Jetter Verlag, 2002
[Kett, 2012]	Kett, Holger: „Referenzmodell zur zielgruppenspezifischen Entwicklung einer webbasierten Informationsplattform für den technischen Vertrieb (IPA-IAO Forschung und Praxis – Nr. 520)"; Jost Jetter Verlag, 2012
[Kleinrock, 1975]	Kleinrock, Leonard: „Queueing Systems – Volume I: Theory"; John Wiley & Sons (Wiley-Interscience), 1975
[Kleinrock, 1976]	Kleinrock, Leonard: „Queueing Systems – Volume II: Computer Applications"; John Wiley & Sons (Wiley-Interscience), 1976
[Knermann, 2007]	Knermann, Christian; Meinecke, Sebastian: „Dünner Kunde – Thin Clients selbst gemacht mit Thinstation", S. 202-205 in „c't Magazin für Computertechnik 3/2007"; Heise Zeitschriften Verlag, 2007
[Knermann, 2007]	Knermann, Christian: „Modellpflege – Citrix Presentation Server 4.5", S. 16-21 in „IT-Administrator Nr. 11/2007"; Heinemann Verlag, 2007
[Knermann, 2008]	Knermann, Christian: „Neuer Lastesel im Netz – Terminaldienste unter Windows Server 2008 (1)", S. 28-32 in „IT-Administrator Nr. 9/2008"; Heinemann Verlag, 2008
[Knermann, 2008a]	Knermann, Christian: „Unsere kleine Farm – Terminaldienste unter Windows Server 2008 (2)", S. 45-50 in „IT-Administrator Nr. 10/2008"; Heinemann Verlag, 2008
[Knermann, 2008b]	Knermann, Christian; Meinecke, Sebastian: „Schlank in allen Belangen – openThinClient als Thin-Client-Alternative", S. 32-37 in „IT-Administrator Nr. 5/2008"; Heinemann Verlag, 2008
[Knermann, 2008c]	Knermann, Christian; Köchling, Christoph: „»PC vs. Thin Client« – Wirtschaftlichkeitsbetrachtung, Version 1.2008"; Fraunhofer-Institut für Umwelt-, Sicherheits- und Energietechnik UMSICHT, 2008
[Knermann, 2009]	Knermann, Christian: „Der Hydra neue Köpfe – Im Test: Citrix XenApp 5.0", S. 14-20 in „IT-Administrator Nr. 3/2009"; Heinemann Verlag, 2009
[Knermann, 2009a]	Knermann, Christian: „Blade Runner – Im Test: IBM CP20 Client und HC10 Workstation Blade", S. 29-34 in „IT-Administrator Nr. 5/2009"; Heinemann Verlag, 2009

Kurzzeichen	Quellenangabe
[Knermann, 2009b]	Knermann, Christian: „Schöner wandern – Roaming Profiles unter Windows 2003/2008", S. 35-39 in „IT-Administrator Nr. 5/2009"; Heinemann Verlag, 2009
[Knermann, 2010]	Knermann, Christian: „Applikationszentrale für Terminalserver – Im Test: Citrix XenApp 6", S. 28-34 in „IT-Administrator Nr. 11/2010"; Heinemann Verlag, 2010
[Knermann, 2010a]	Knermann, Christian: „Arbeitsplatz im Netz – Im Test: Citrix XenDesktop 4.0", S. 12-18 in „IT-Administrator Nr. 4/2010"; Heinemann Verlag, 2010
[Knermann, 2010b]	Knermann, Christian: „Systeme am laufenden Band – Virtuelle Maschinen mit dem Citrix Provisioning Server warten (1)", S. 45-47 in „IT-Administrator Nr. 4/2010"; Heinemann Verlag, 2010
[Knermann, 2010c]	Knermann, Christian: „Updates am laufenden Band – Virtuelle Maschinen mit dem Citrix Provisioning Server warten (2)", S. 44-46 in „IT-Administrator Nr. 5/2010"; Heinemann Verlag, 2010
[Knermann, 2010d]	Knermann, Christian: „Massenware Desktop – Virtuelle Maschinen mit dem Citrix Provisioning Server warten (3)", S. 44-46 in „IT-Administrator Nr. 6/2010"; Heinemann Verlag, 2010
[Knermann, 2011]	Knermann, Christian: „Applikationen im Fernzugriff – Microsoft RemoteApps einrichten", S. 36-40 in „IT-Administrator Nr. 5/2011"; Heinemann Verlag, 2011
[Knermann, 2011a]	Knermann, Christian: „Runderneuerte Desktop-Schmiede – Im Test: Citrix XenDesktop 5", S. 24-29 in „IT-Administrator Nr. 5/2011"; Heinemann Verlag, 2011
[Knermann, 2013]	Knermann, Christian: „Desktop(r)evolution – Im Test: Citrix XenDesktop 7", S. 12-17 in „IT-Administrator Nr. 12/2013"; Heinemann Verlag, 2013
[Knermann, 2014]	Knermann, Christian: „Hypervisor to go – Im Test: Citrix XenClient 5", S. 14-18 in „IT-Administrator Nr. 1/2014"; Heinemann Verlag, 2014
[Knermann, 2014a]	Knermann, Christian: „Virtuelle Integration – Neue Remote Desktop Services in Windows Server 2012 R2", S. 38-41 in „IT-Administrator Nr. 1/2014"; Heinemann Verlag, 2014
[Knermann, 2015]	Knermann, Christian: „Weltoffen – Linux-Systeme in Citrix XenDesktop 7.x integrieren", S. 40-45 in „IT-Administrator Nr. 10/2015"; Heinemann Verlag, 2015
[Knöpfel, 2004]	Knöpfel, Andreas: „Konzepte der Beschreibung interaktiver Systeme"; Hasso-Plattner-Institut für Softwaresystemtechnik an der Universität Potsdam, 2004
[Köchling, 2004]	Köchling, Christoph: „Kostenmodell Server Based Computing – Spezifische Beschaffungs- und Betriebskosten"; Fraunhofer-Institut für Umwelt-, Sicherheits- und Energietechnik UMSICHT, 2004

Kurzzeichen	Quellenangabe
[Königsmann, 2011]	Königsmann, Kay-Uwe: „Kontrolldienste – Application-Whitelisting-Ansätze im Vergleich", S. 56-61 in „iX Magazin für professionelle Informationstechnik 5/2011", Heise Zeitschriften Verlag, 2011
[Kofler, 1999]	Kofler, Michael: „Linux – Installation, Konfiguration, Anwendung"; Addison-Wesley-Longman, 1999
[Kruse, 1996]	Kruse, Christian: „Referenzmodellgestütztes Geschäftsprozeßmanagement – Ein Ansatz zur prozeßorientierten Gestaltung vertriebslogistischer Systeme"; Gabler Verlag, 1996
[Kugel, 2005]	Kugel, Rudolf: „Ein Beitrag zur Problematik der Integration virtueller Maschinen"; Hasso-Plattner-Institut für Softwaresystemtechnik an der Universität Potsdam, 2005
[Lazowska, 1984]	Lazowska, Edward D.; Zahorjan, John; Graham, G. Scott; Sevcik, Kenneth: „Quantitiave System Performance – Computer System Analysis Using Queueing Network Models"; Prentice-Hall, 1984
[Leemhuis, 2014]	Leemhuis, Thorsten: „Hafenarbeiter – Linux: Container-Virtualisierung mit Docker", S. 146-151 in „c't Magazin für Computertechnik 17/2014"; Heise Zeitschriften Verlag, 2014
[Lenz, 2010]	Lenz, Daniel J.: „Paravirtualisierung und Security am Beispiel Xen"; Research Group Systems Modelling – Institute for Computer Science and Business Information Systems – Universität Duisburg-Essen, 2010
[Lenz, 2010a]	Lenz, Daniel J.: „Ein prototypischer Xen-Cluster zur Sicherheitsoptimierung in virtualisierten Rechenzentren"; Research Group Systems Modelling – Institute for Computer Science and Business Information Systems – Universität Duisburg-Essen, 2010
[Lüdemann, 2007]	Lüdemann, Nico: „Applikationsvirtualisierung mit Microsoft SoftGrid 4"; Galileo Computing, 2007
[Lüdemann, 2009]	Lüdemann, Nico: „Citrix XenApp 5 – Das Praxishandbuch für Administratoren"; Galileo Computing, 2009
[Lüdemann, 2011]	Lüdemann, Nico: „Citrix XenApp 6 und XenDesktop 5: Das Praxishandbuch für Administratoren"; Galileo Computing, 2011
[Luxem, 2000]	Luxem, Redmer: „The Impact of Trading Digital Products on Retail Information Systems", in „Proceedings of the 33rd Annual Hawaii International Conference on System Sciences – 2000"; IEEE Computer Society, 2000
[Madden, 2004]	Madden, Brian S.; Oglesby, Ron: „Terminal Services for Microsoft Windows Server 2003 – Advanced Technical Design Guide"; BrianMadden.com Publishing, 2004

Kurzzeichen	Quellenangabe
[Martius, 2000]	Martius, Kai: „Sicherheitsmanagement in TCP/IP-Netzen – Aktuelle Protokolle, praktischer Einsatz, neue Entwicklungen", Friedr. Vieweg und Sohn, 2000
[Mathers, 2000]	Mathers, Todd W.: „Windows NT/2000 Thin Client Solutions – Implementing Terminal Services and Citrix MetaFrame"; New Riders, 2000
[Mauro, 2005]	Mauro, Douglas R.; Schmidt, Kevin J.: „Essential SNMP, Second Edition"; O'Reilly Media, Inc., 2005
[Menascé, 2004]	Menascé, Daniel A.; Almeida, Virgilio A. F.; Dowdy, Lawrence W.: „Performance by Design – Computer Capacity Planning by Example"; Pearson Education/Prentice-Hall PTR, 2004
[Mucha, 2004]	Mucha, M.: „Referenzmodell für ein Produktdaten Clearing Center – am Beispiel eines Informationsmodells für die Elektrowirtschaft (IPA-IAO Forschung und Praxis – Nr. 403)"; Jost Jetter Verlag, 2004
[Müller-C., 2001]	Müller-Clostermann, Bruno (Hrsg.): „Kursbuch Kapazitätsmanagement – Kompendium für Planung, Analyse und Tuning von IT-Systemen"; Institut für Informatik der Universität Essen, 2001
[Müller-C., 2005]	Müller-Clostermann, Bruno: „Capacity Planning and Performance Evaluation with the Tool VITO 2.0"; Institut für Informatik und Wirtschaftsinformatik der Universität Duisburg-Essen, 2005
[Müller-C., 2012]	Müller-Clostermann, Bruno: „Stochastische Netze 1"; Institut für Informatik und Wirtschaftsinformatik der Universität Duisburg-Essen, 2012
[Nye, 1994]	Nye, Adrian: „Definitive Guides to the X Window System – Volume One: Xlib Programming Manual"; O'Reilly & Associates, Inc., 1994
[Odom, 2008]	Odom, Wendell: „CCNA ICND2 Official Exam Certification Guide, Second Edition"; Cisco Press, 2008
[Plura, 2014]	Plura, Michael: „Im Terminal – Ressourcenschonende virtuelle Maschinen mit Containern betreiben", S. 110-115 in „iX Magazin für professionelle Informationstechnik 6/2014", Heise Zeitschriften Verlag, 2014
[Popek, 1974]	Popek, Gerald J.; Goldberg, Robert P.: „Formal requirements for virtualizable third generation architectures", S. 412-421 in „Communications of the ACM – Volume 17 Issue 7, July 1974" ; Association for Computing Machinery, 1974

Kurzzeichen	Quellenangabe
[Porter, 2011]	Porter, Donald E.; Boyd-Wickizer, Silas; Howell, Jon; Olinsky, Reuben; Hunt, Galen: „Rethinking the Library OS from the Top Down", S. 291-304 in „Proceedings of the 16th International Conference on Architectural support for programming languages and operating systems (ASPLOS)"; Association for Computing Machinery, 2011
[Remmert, 2001]	Remmert, Jan: „Referenzmodellierung für die Handelslogistik"; Deutscher Universitäts-Verlag, 2001
[Rhenau, 2005]	Rhenau, Wieland: „Seminar ‚Ausgewählte Themen des Software Engineering' – Fundamental Modeling Concepts (Präsentation)"; Institut für Informatik der Freien Universität Berlin, 2005
[Rutkowska, 2010]	Rutkowska, Joanna; Wojtczuk, Rafal: „Qubes OS Architecture – Version 0.3"; Invisble Things Lab, 2010
[Robin, 2000]	Robin, John Scott; Irvine, Cynthia E.: „Analysis of the Intel Pentium's Ability to Support a Secure Virtual Machine Monitor", S. 10ff. in „Proceedings of the 9th conference on USENIX Security Symposium – Volume 9 (SSYM'00)"; USENIX Association Berkeley, 2000
[SAP, 2007]	SAP AG (Hrsg.): „Standardized Technical Architecture Modeling – Conceptual and Design Level (Version 1.0)"; SAP AG, 2007
[Schmidt, 2016]	Schmidt, Jürgen: „Entschlüsselt – Der Krypto-Wegweiser für Nicht-Kryptologen", S. 174-177 in „c't Magazin für Computertechnik 1/2016"; Heise Zeitschriften Verlag, 2016
[Schweitzer, 1979]	Schweitzer, Paul J.: „Approximate analysis of multi-class closed networks of queues", in „Proceedings of the International Conference on Stochastic Control and Optimization"; Free University Amsterdam, 1979
[SCOR, 2010]	Supply Chain Council (Hrsg.): „Supply Chain Council: SCOR 10.0 Overview Booklet"; Supply Chain Council, 2010
[Söldner, 2012]	Söldner, Jens-Henrik; Ewald, Mathias: „Matroschka 2.0 – Verschachtelte VMs mit heterogenen Systemen", S. 142-145 in „iX Magazin für professionelle Informationstechnik 4/2012"; Heise Zeitschriften Verlag, 2012
[Spruijt, 2013]	Spruijt, Ruben: „VDI Smackdown v2.3"; PQR B. V., 2013
[Spruijt, 2013a]	Spruijt, Ruben: „Application Virtualization Smackdown v4.1"; PQR B. V., 2013
[Thomas, 2006]	Thomas, Oliver: „Das Referenzmodellverständnis in der Wirtschaftsinformatik: Historie, Literaturanalyse und Begriffsexplikation" in „IWi Heft 187"; Institut für Wirtschaftsinformatik an der Universität des Saarlandes, 2006

Kurzzeichen	Quellenangabe
[Totzauer, 1987]	Totzauer, Günter: „Kopplung der Kettendurchsätze in geschlossenen Warteschlangennetzwerken", S. 360-374 in „Messung, Modellierung und Bewertung von Rechensystemen, Informatik-Fachberichte Volume 154"; Springer Verlag, 1987
[Tritsch, 2003]	Tritsch, Dr. Bernhard: „Microsoft Windows Server 2003 Terminaldienste"; Microsoft Press, 2003
[Tulloch, 2010]	Tulloch, Mitch: „Understanding Microsoft Virtualization Solutions – From the Desktop to the Datacenter, Second Edition"; Microsoft Press, 2010
[Yu, 2006]	Yu, Yang; Guo, Fanglu; Nanda, Susanta; Lam, Lap-chung; Chiueh, Tzi-cker: „A Feather-weight Virtual Machine for Windows Applications", S. 24-34 in „Proceedings of the 2nd international conference on Virtual execution environments (VEE '06)"; Association for Computing Machinery, 2006
[Yu, 2007]	Yu, Yang: „OS-level Virtualization and Its Applications"; State University of New York at Stony Brook, 2007
[Yu, 2008]	Yu, Yang; Govindarajan, Harhiharahn Kolam; Lam, Lap-chung; Chiueh, Tzi-cker: „Applications of a Feather-weight Virtual Machine", S. 171-180 in „Proceedings of the fourth ACM SIGPLAN/SIGOPS international conference on Virtual execution environments (VEE '08)"; Association for Computing Machinery, 2008
[Vogel, 2010]	Vogel, Robert; Koçoğlu, Tarkan; Berger, Thomas: „Desktopvirtualisierung: Definitionen – Architekturen – Business-Nutzen"; Vieweg+Teubner Verlag, 2010
[Watts, 2011]	Watts, David; Davis, Randall; Kroutov, Ilia: „IBM BladeCenter Products and Technology – Eighth Edition (July 2011)"; IBM, 2011
[Whitaker, 2002]	Whitaker, Andrew; Shaw, Marianne; Gribble, Steven D.: „Denali: Lightweight Virtual Machines for Distributed and Networked Applications"; University of Washington, 2002
[Whitaker, 2002a]	Whitaker, Andrew; Shaw, Marianne; Gribble, Steven D.: „Denali: A Scalable Isolation Kernel", S. 10-15 in „Proceedings of the 10th workshop on ACM SIGOPS European workshop (EW 10)"; Association for Computing Machinery, 2002
[Whitaker, 2002b]	Whitaker, Andrew; Shaw, Marianne; Gribble, Steven D.: „Scale and Performance in the Denali Isolation Kernel", S. 195-209 in „ACM SIGOPS Operating Systems Review - OSDI '02: Proceedings of the 5th symposium on Operating systems design and implementation – Volume 36 Issue SI, Winter 2002"; Association for Computing Machinery, 2002
[Zeichick, 2006]	Zeichick, Alan: „Processor-Based Virtualization, AMD64 Style, Part I+II"; Advanced Micro Devices, 2006

Kurzzeichen	Quellenangabe
[Zimmer, 2006]	Zimmer, Dennis: „Professionelle Virtualisierungsprodukte: ein Überblick – Trau, schau, wem", S. 64-70 in „iX Magazin für professionelle Informationstechnik 5/2006", Heise Zeitschriften Verlag, 2006
[Zimmermann, 1980]	Zimmermann, Hubert: „OSI Reference Model – The ISO Model of Architecture for Open Systems Interconnection", S. 425-432 in „IEEE Transactions on Communications, vol. 28, no. 4, April 1980"; IEEE Computer Society, 1980

10 Appendix A: FMC Blockdiagramme

Wie in Kapitel 5.3.3 erläutert, eignen sich die **Fundamental Modeling Concepts (FMC)** zur Beschreibung der Systemstrukturen eines zu modellierenden Sachverhalts, indem sie auf dessen Architektur und Aufbau fokussieren. Die FMC unterscheiden dazu die drei verschiedenen Sichten der *kompositionellen Struktur*, der *dynamischen Struktur* und der *Wertebereichsstruktur*, wobei im vorliegenden Fall insbesondere die kompositionelle Struktur relevant ist. Dieses Kapitel erläutert die Syntax der im Kontext der kompositionellen Struktur verwendeten Blockdiagramme.

Bei einem FMC Blockdiagramm handelt es sich um einen sogenannten bipartiten oder paaren Graphen. Dies bedeutet, dass sich sämtliche Knoten des Graphen derart auf zwei disjunkte Teilmengen A und S verteilen, dass innerhalb einer der beiden Teilmengen keine Knoten direkt über Kanten des Graphen miteinander verbunden sind. Ein Knoten der Menge A darf also nur mit einem oder mehreren Knoten der Menge S verbunden sein und umgekehrt darf ein Knoten der Menge S nur mit einem oder mehreren Knoten der Menge A verbunden sein.

10.1 Sprachelemente

Die Syntax der Blockdiagramme unterscheidet die Teilmenge A der aktiven Systemkomponenten und die Teilmenge S der passiven Komponenten, auch Lokationen genannt.

Abbildung 10-1: Akteure

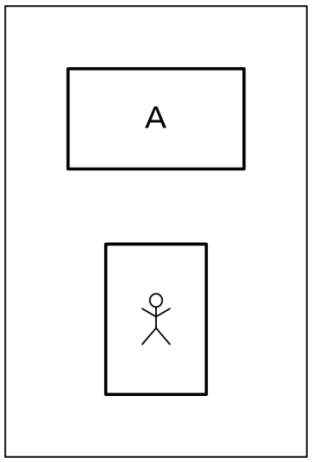


Abbildung 10-2: Speicher/Kanäle

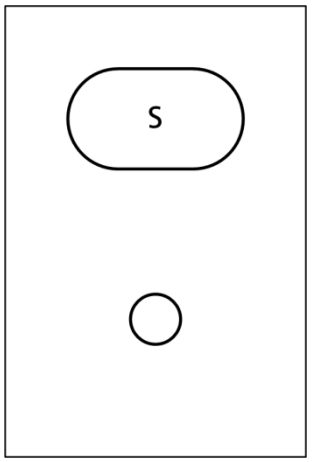
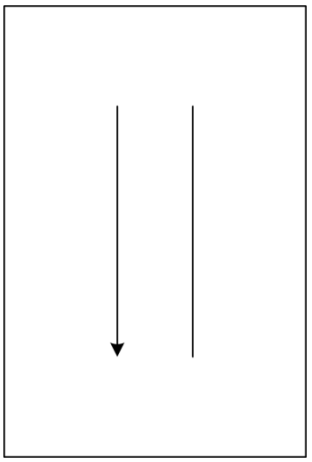


Abbildung 10-3: Kanten



Quelle: Eigene Darstellungen (nach [Apfelbacher, 2005a][108])

[108] Das „Reference Sheet" zur Syntax der FMC Blockdiagramme ist verfügar unter der folgenden Adresse: http://www.fmc-modeling.org/download/notation_reference/Reference_Sheet-Block_Diagram.pdf (abgerufen am 01.02.2016).

Aktive Komponenten werden als Agenten oder Akteure bezeichnet und durch eckige Formen symbolisiert (Abbildung 10-1). Passive Komponenten umfassen Speicher, die durch Formen mit abgerundeten Ecken dargestellt werden, sowie Kanäle, symbolisiert durch Kreise (Abbildung 10-2). Gerichtete oder ungerichtete Kanten (Abbildung 10-3) verbinden die Knoten des Graphen miteinander und spezifizieren, wie aktive auf passive Systemkomponenten zugreifen. Hierbei sind lesender oder schreibender Zugriff sowie die Kombination beider Zugriffe zu unterscheiden.

Abbildung 10-4: Lesen

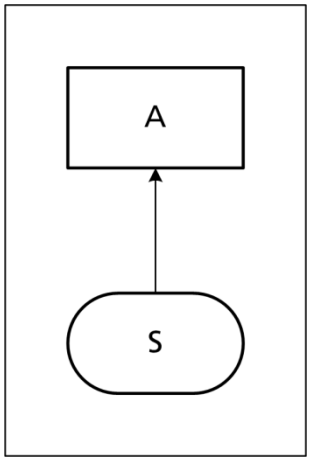


Abbildung 10-5: Schreiben

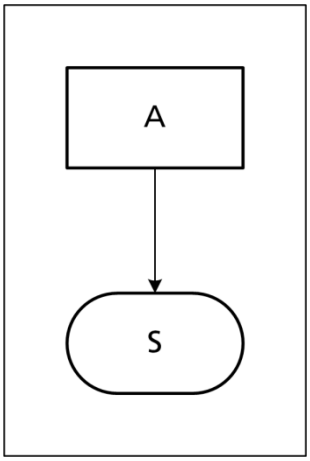
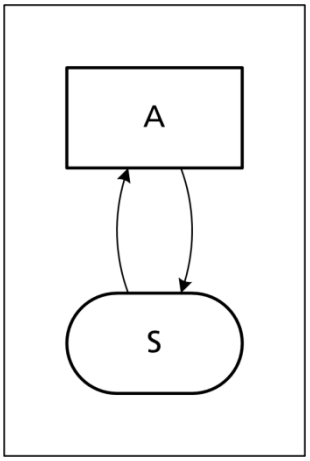


Abbildung 10-6: Lesen/Schreiben



Quelle: Eigene Darstellungen (nach [Apfelbacher, 2005a])

Bei lesendem Zugriff weist die gerichtete Kante vom Speicher zum Akteur (Abbildung 10-4), bei schreibendem Zugriff vom Akteur zum Speicher (Abbildung 10-5). Bei der Kombination von Lese- und Schreib-Operationen werden entsprechend zwei gerichtete Kanten notiert (Abbildung 10-6).

Abbildung 10-7: Bidir. Kanal

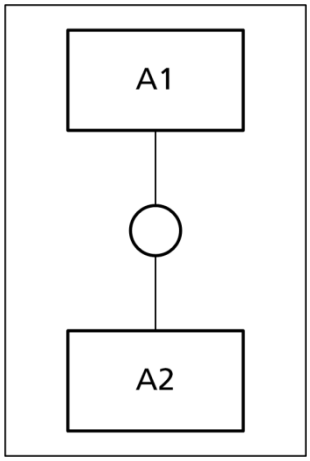


Abbildung 10-8: Unidir. Kanal

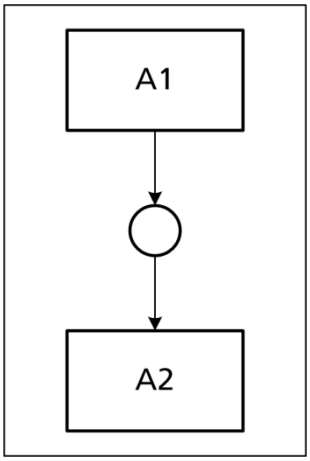
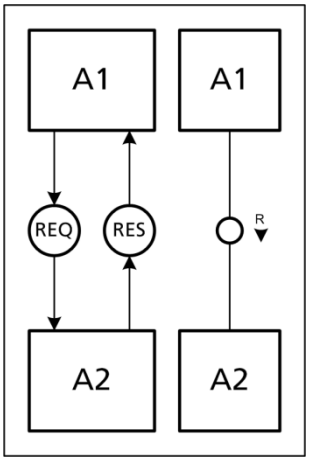


Abbildung 10-9: Request-Response



Quelle: Eigene Darstellungen (nach [Apfelbacher, 2005a])

Die ungerichtete Kante kommt i. d. R. bei Kanälen zum Einsatz, um bidirektionale Kommunikation zweier Akteure über den Kanal darzustellen (Abbildung 10-7), während bei der unidirektionalen Kommunikation über einen Kanal zwei gerichtete Kanten notiert werden. So kann beispielsweise in Abbildung 10-8 der Akteur $A1$ Informationen an Akteur $A2$ senden, umgekehrt ist dies jedoch nicht möglich.

Für den Fall von Kommunikation nach dem Request-Response-Prinzip existieren zwei Varianten der Notation. In Abbildung 10-9 sendet der Akteur $A1$ eine Anfrage an Akteur $A2$, der diese entsprechend beantwortet. In der ausführlichen Variante werden Sende- und Antwort-Kanal separat notiert. In vereinfachter Notation wird neben einem ungerichteten Kanal die Sende-/ Empfangsrichtung durch einen Pfeil angezeigt. Zusätzlich kann nach Bedarf neben dem Kanal das verwendete Netzwerkprotokoll oder der verwendete Port notiert werden.

Als Alternative zu Kanälen können Akteure Informationen auch über einen gemeinsamen Speicherbereich austauschen. Dieser Fall wird notiert, indem zwei oder mehr Akteure jeweils Lese-/Schreib-Operationen auf diesem Speicher ausführen (Abbildung 10-10).

Abbildung 10-10: Gemeinsamer Speicher

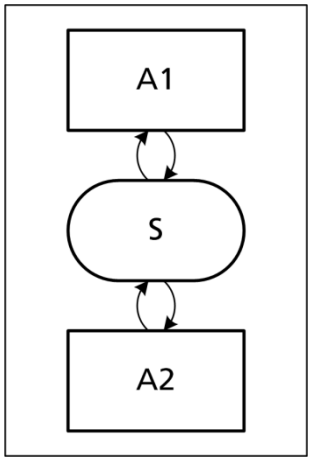
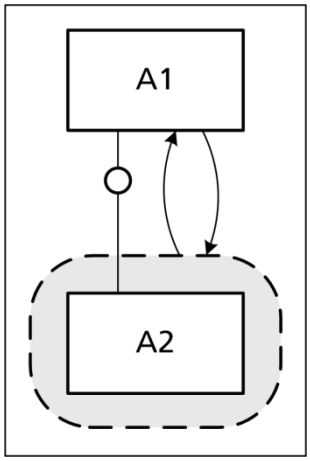


Abbildung 10-11: Dynamische Veränderung



Quelle: Eigene Darstellungen (nach [Apfelbacher, 2005a])

Weiterhin erlaubt es die Syntax der FMC Blockdiagramme, dynamische Veränderungen in der Struktur eines Systems darzustellen. Veränderungen zur Laufzeit, die einer Infrastruktur neue Elemente hinzufügen oder bestehende Elemente entfernen, werden durch Bereiche mit einer gestrichelten Linie als Umrandung und abgerundeten Ecken dargestellt.

In Abbildung 10-11 erzeugt und/oder löscht der Akteur $A1$ einen dynamischen Bereich, welcher den Akteur $A2$ enthält. Nachdem Akteur $A1$ den Akteur $A2$ erzeugt hat, können beide aktiven Komponenten über einen bidirektionalen Kanal miteinander kommunizieren. Im Kontext von Server Based Computing Infrastrukturen eignet sich dieses Sprachelement beispielsweise, um die dynamische Bereitstellung von Terminal Servern oder virtuellen Desktops mittels Machine Creation Services (MCS) oder Provisioning Services (PVS) abzubilden (vgl. Kapitel 2.3.1 und 5.4.3.4).

10.2 Empfehlungen zur Verwendung

Neben den zuvor genannten Sprachelementen der FMC Blockdiagramme geben die „FMC Visualization Guidelines“ (vgl. [Apfelbacher, 2005]) weitere Handreichungen zur Verwendung der Sprache. Diese haben teils den Charakter von *Empfehlungen*, die Anwendung finden können, und nicht von *Regeln*, die eingehalten werden müssen.

Die Autoren unterscheiden dabei Struktur und Layout eines Graphen. Die Struktur bezeichnet den eigentlichen inhaltlichen Aspekt, also welche Elemente des Graphen wie miteinander verbunden sind und somit interagieren. Ein und dieselbe Struktur kann grafisch auf verschiedene Weisen dargestellt werden – abhängig davon, wie die Knoten des Graphen auf der Zeichenfläche angeordnet werden. Das Layout sollte dabei Wahrnehmung und Verständnis des dargestellten Sachverhalts unterstützen.

So sollten sich die Linienbreiten von Knoten und Kanten unterscheiden, so dass diese einfacher voneinander zu unterscheiden sind. Kanten sollten weiterhin ausschließlich horizontal oder vertikal verlaufen und sich möglichst nicht überschneiden. Auch sollten Knoten Kanten nicht überlagern. Weiterhin wird empfohlen, Kanten immer mit abgerundeten Ecken darzustellen und Kanten mit dem gleichen Ziel zu Bäumen zusammenzufassen. Bei der Beschriftung sollte eine einheitliche Schriftart oder -familie Verwendung finden.

10.3 Weitere Sprachelemente

Farben und Gruppierungen können zum Einsatz kommen, um zusammengehörige Elemente eines Graphen von anderen abzugrenzen. So stellt Abbildung 10-12 jeweils Lese-/Schreib-Zugriffe eines Akteurs A auf zwei Speicher $S1$ und $S2$ dar. Abbildung 10-13 fasst diese Speicher zu einer Gruppe zusammen, so dass die Lese-/Schreib-Zugriffe nur noch einmal notiert werden müssen.

Abbildung 10-12: Beispiel für Lese-/Schreib-Zugriffe

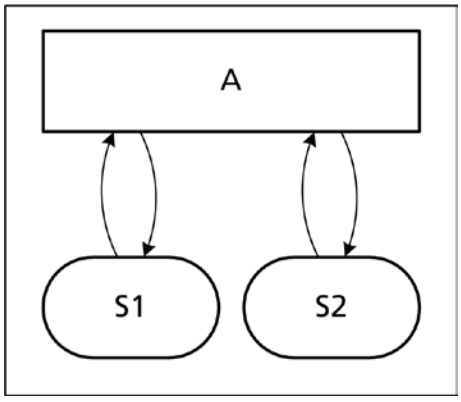
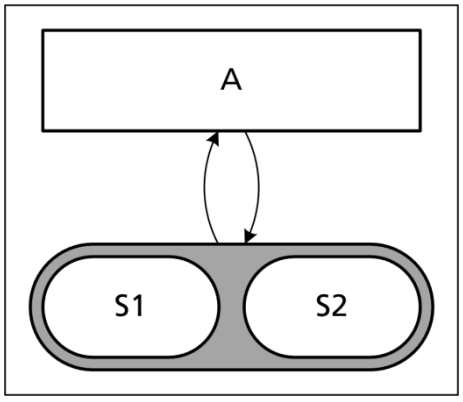


Abbildung 10-13: Gruppierung von Elementen



Quelle: Eigene Darstellungen (nach [Apfelbacher, 2005], S. 24)

Weiterhin führen die „FMC Visualization Guidelines“ eine Notation zur Enumeration mehrerer gleichartiger Komponenten ein. So kommuniziert in Abbildung 10-14 ein Akteur A mit einer Gruppe von Akteuren B_1 bis B_n. Dieser Sachverhalt ließe sich ebenso durch die in Abbildung 10-15 gewählte Notation abbilden.

Abbildung 10-14: Enumeration, Variante 1

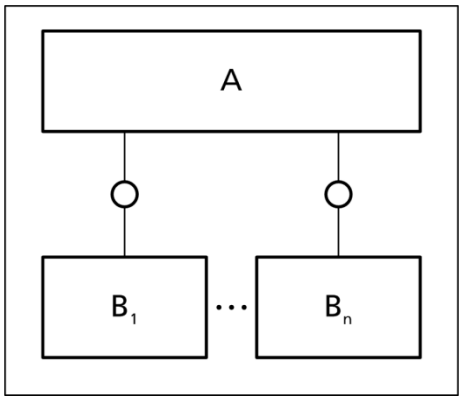
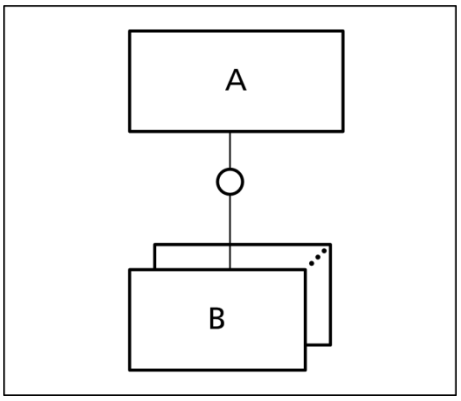


Abbildung 10-15: Enumeration, Variante 2



Quelle: Eigene Darstellungen (nach [Apfelbacher, 2005], S. 25)

Beide Notationen sind gleichwertig zu verwenden und es gibt keine strikte Regel, die vorschreibt, welcher der beiden Varianten der Vorzug zu geben ist. Die erste Variante ist besser geeignet, wenn die Anzahl $1 \ldots n$ der Akteure B fix oder von besonderer inhaltlicher Relevanz ist. Die zweite Variante eignet sich für eine unbestimmte Anzahl von Akteuren und ist bezüglich der grafischen Darstellung platzsparender.

Wie das zuvor erläuterte Sprachelement zur Darstellung dynamischer Veränderungen ist die Notation der Enumeration besonders im Kontext von Server Based Computing Infrastrukturen nützlich, um den Umgang mit zahlreichen Terminal Servern oder virtuellen Desktops darzustellen, die mittels Machine Creation Services (MCS) oder Provisioning Services (PVS) verwaltet werden. Beispiele für die Verwendung finden sich in den Kapiteln 5.4.3.3 bis 5.4.3.5.

11 Appendix B: Grundlagen stochastischer Modelle

Das in Kapitel 6 aus dem Referenzmodell abgeleitete VITO-Modell einer SBC Infrastruktur basiert auf stochastischen Modellen. Die Analytik unterstützt die Konzeption und Weiterentwicklung von IT-Systemen und ist nützlich, in vielen Fällen sogar unumgänglich, um komplexe Systeme und die Zusammenhänge ihrer Komponenten zu erfassen und einer strukturierten Herangehensweise beim Kapazitätsmanagement zugänglich zu machen. Diese Hintergründe wurden in Kapitel 5.4.4.2 erläutert, die Grundlagen der Modelle, die auch die Basis für die in der Software VITO implementierten Algorithmen bilden, werden im Folgenden kurz zusammengefasst und es werden zur weiteren Vertiefung relevante Literaturstellen empfohlen.

11.1 Markov-Ketten

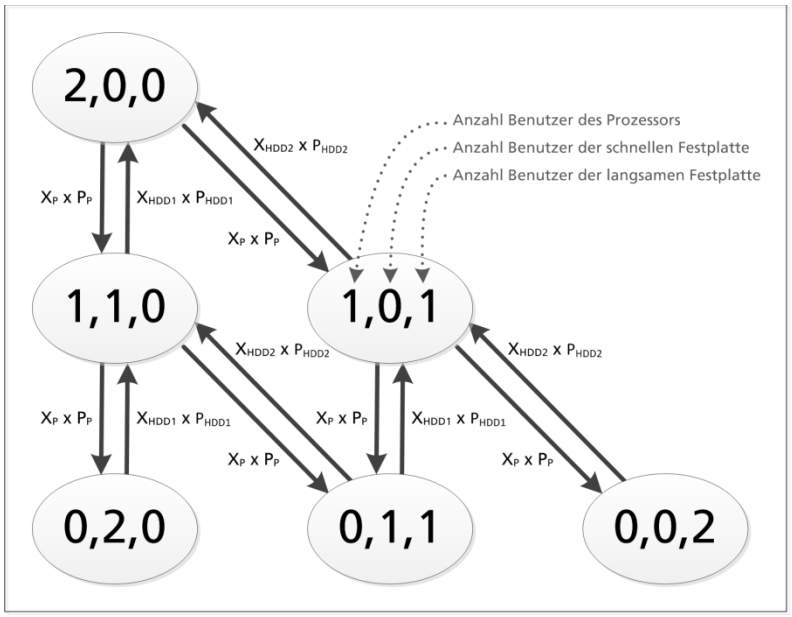
Die sogenannten zustandsdiskreten Markov-Ketten sind eine spezielle Form gerichteter Graphen, welche die diskreten Zustände, in denen sich ein zu modellierendes Systemen befinden kann, als Knoten abbilden, während die möglichen Übergänge zwischen diesen Knoten durch Zufallsvariable dargestellt werden. Die Übergänge zwischen den Zuständen sind also nicht deterministisch, sondern *können* mit einer bestimmten Wahrscheinlichkeit eintreten, *müssen* dies jedoch nicht zwingend. Dieser Ansatz eignet sich damit sehr gut, um das in der Praxis ebenfalls nicht deterministische Verhalten einer größeren Zahl von Arbeitslasten (z. B. Benutzer, Rechenaufträge, Datenbankabfragen) in einem komplexen IT-Systemen zu modellieren (vgl. [Müller-C., 2012], S. 12ff.).

Die Modellierung beginnt entsprechend mit der Spezifikation sämtlicher Zustände, in denen sich das zu untersuchende System befinden kann. Im nächsten Schritt werden die möglichen Übergänge spezifiziert und schlussendlich für zeitdiskrete Ketten („Discrete-Time Markov Chains“ (vgl. [Kleinrock, 1975], S. 26ff.)) mit Wahrscheinlichkeiten für den Wechsel von einem Zustand in den anderen bzw. Raten im Falle von zeitkontinuierlichen Ketten („Continuous-Time Markov Chains“ (vgl. [Kleinrock, 1975], S. 44ff.)) parametrisiert. Ist das Modell in dieser Weise aufgebaut, kann es *gelöst* werden, wobei diese Lösung darin besteht, anhand des Zustandsdiagramms ein lineares Gleichungssystem zu bilden und dieses im Hinblick auf die langfristig stabilen Zustandswahrscheinlichkeiten – im Englischen *steady state probabilities* – aufzulösen.

Dieser Ansatz, der zunächst abstrakt erscheinen mag, sei erläutert angelehnt an eines der Beispiele, die Menascé et. al verwendeten, um die Modellierungstechnik zu verdeutlichen (vgl. [Menascé, 2004], S. 260ff.): Zwei Benutzer arbeiten mit einem IT-System bestehend aus einem Prozessor sowie einer langsamen und einer schnellen Festplatte (doppelt so schnell wie die erste). Jede Transaktion, die von einem Benutzer ausgelöst wird, nutzt abwechselnd den Prozessor sowie Dateien auf einer der beiden Festplatten, wobei die Wahrscheinlichkeit, die gewünschte Datei auf dieser oder jener Festplatte zu finden, gleich verteilt ist. Ausgehend von

diesen Annahmen gelangten die Autoren zu einem Diagramm aller sechs möglichen Zustände dieses Systems als Knoten und den Zustandsübergängen als Kanten, die mit den den Durchsätzen von Prozessor (X_P), Festplatte 1 (X_{HDD1}) und Festplatte 2 (X_{HDD2}) sowie den Übergangswahrscheinlichkeiten zwischen den Zuständen (P_P, P_{HDD1}, P_{HDD2}) parametrisiert werden (vgl. Abbildung 11-1), was zu einem linearen Gleichungssystem führt, das wiederum benutzt werden kann, um die *steady state probabilities* zu berechnen – also die langfristigen Wahrscheinlichkeiten, das System zu einem beliebigen Beobachtungszeitpunkt in einem der Zustände vorzufinden.

Abbildung 11-1: Diagramm einer Markov-Kette mit sechs Zuständen



Quelle: Eigene Darstellung, nach [Menascé, 2004], S. 268

Wie Menascé et. al zeigen, ist ein aus einer Markov-Kette resultierendes Gleichungssystem mit N Gleichungen und N Variablen jedoch zunächst unterdeterminiert, da sich immer eine Gleichung aus den übrigen herleiten lässt (vgl. [Menascé, 2004], S. 270). Im zuvor referenzierten Beispiel lässt sich mittels der Normalisierungsbedingung dennoch eine Lösung finden, indem eine der Gleichungen durch die Summe der Wahrscheinlichkeiten aller Zustände ersetzt wird, für diesen konkreten Fall also:

Formel 11-1: Normalisierungsbedingung (nach [Menascé, 2004], S. 272)

$$P_{(2,0,0)} + P_{(1,1,0)} + P_{(1,0,1)} + P_{(0,1,1)} + P_{(0,2,0)} + P_{(0,0,2)} = 1.0$$

Und in allgemeiner Form für ein Modell mit N möglichen Zuständen:

Formel 11-2

$$\sum_{1}^{N} P_N = 1.0$$

Dies ist möglich auf Grund der Annahme, dass es sich um ein geschlossenes System handelt, dass sich das System also zu jedem Zeitpunkt in genau einem der diskreten Anzahl N möglicher Zustände befindet. Eine weitere zentrale Annahme für diese Art der Modellierung ist die sogenannte *Gedächtnislosigkeit*, d. h. man nimmt an, dass die Vorgeschichte des Systems irrelevant für seinen gegenwärtigen Zustand und seine Zukunft ist: „(…) the Markov property insists that the past history be completely summarized in the specification of the current state (…)" (vgl. [Kleinrock, 1975], S. 22). Sämtliche Informationen zur Beschreibung des Systems und seiner weiteren Entwicklung, insbesondere in welchen Zustand das System im nächsten Schritt wechselt, sind ausschließlich abhängig vom aktuellen Zustand.

Dem Vorteil der einfachen Lösbarkeit von Markov-Ketten steht als Nachteil jedoch die Größe des Zustandsraums als begrenzender Faktor gegenüber, denn je mehr unterschiedliche Zustände möglich sind, desto komplexer wird das Zustandsdiagramm. So würde bereits die Erweiterung des vorangegangenen Beispiels um nur einen weiteren Benutzer die Anzahl der Zustände und damit der Gleichungen und Variablen auf zehn mögliche Zustände vergrößern.

Die Betrachtung umfangreicherer Systeme mit einer deutlich größeren Anzahl von K Komponenten und einer wesentlich höheren Anzahl von N Benutzern[109] würde entsprechend der Kombinatorik zu einer Explosion des Zustandsraums auf eine Anzahl von

Formel 11-3

$$\binom{N+K-1}{K-1} = \frac{(N+K-1)!}{(K-1)!\,\big((N+K-1)-(K-1)\big)!} = \frac{(N+K-1)!}{(K-1)!\,N!}$$

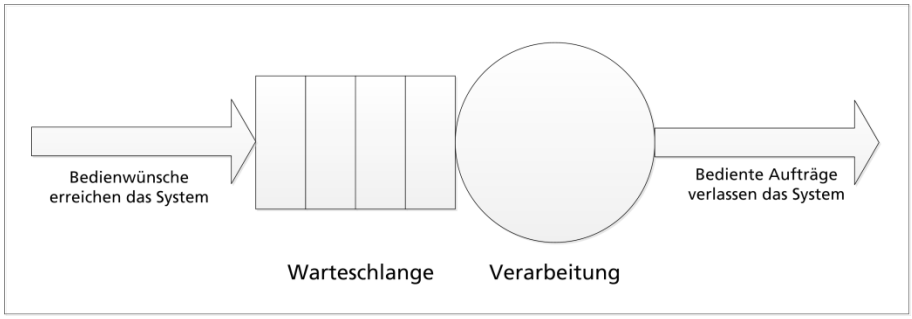
möglichen Zuständen führen und eine Abbildung und Lösung des Modells kaum mehr beherrschbar werden lassen. Bei der Untersuchung komplexerer Systeme sind also anderweitige Methoden erforderlich, um eine Lösung zu approximieren. Bevor ein Ansatz hierzu erläutert wird, werden jedoch zunächst die Begriffe der Warteschlangen und Warteschlangennetze eingeführt.

11.2 Warteschlangen(netze)

Eine einfache und intuitive Darstellung von Gegebenheiten der realen Welt (z. B. Kunden an der Supermarkt-Kasse oder an einem Bankautomaten) ist das Prinzip einer einfachen Warteschlange, bei der eine Verarbeitungseinheit zu einem Zeitpunkt genau einen Bedienwunsch bearbeitet (Abbildung 11-2). Neue Bedienwünsche erreichen das System und warten in einem Warteraum darauf, dass ein bedienter Auftrag das System verlässt und sie selbst von der Verarbeitungseinheit angenommen werden (vgl. „Single Service Center" in [Lazowska, 1984], S. 4 ff.).

[109] Unter den Annahmen, dass es sich um eine fixe Zahl identischer Benutzer handelt und dass der Bedienwunsch eines jeden Benutzers an jede Komponente exponential verteilt ist (vgl. [Menascé, 2004], S. 311).

Abbildung 11-2: Einfache Warteschlange



Quelle. Eigene Darstellung, nach [Lazowska, 1984], S. 5

Es liegt nahe, dieses Prinzip auch auf das Feld der IT-Systemtechnik zu übertragen. An die Stelle der abtrakten Verarbeitungseinheit treten dann konkrete Komponenten, wie Prozessor oder Festplatte eines Systems, oder bei einem niedrigeren Detaillierungsgrad ganze Server, wie Datei-, Datenbank- oder eben die in Kapitel 6 exemplarisch betrachteten Terminal Server.

11.2.1 Offene und geschlossene Klassen

Sofern verschiedene Kunden mit unterschiedlichen Bedienwünschen existieren, werden diese in *Klassen* unterteilt. Die Klassen wiederum werden, wie auch die Wartenschlange als Ganzes, als *offen* oder *geschlossen* bezeichnet (vgl. [Menascé, 2004], S. 40ff). In ersterem Fall erreichen Kunden von *außen* die Warteschlange und verlassen anschließend wieder das System. Dies bedeutet, dass die Anzahl der Kunden und somit Bedienwünsche im System über die Zeit variabel ist und die Last von der Ankunftsrate der Kunden abhängt. In letzterem Fall kursiert eine über die Zeit konstante Anzahl von Kunden im System und verlässt dieses nicht, so dass die Last durch die Anzahl ihrer Bedienwünsche spezifiziert ist.

11.2.2 Bedienstrategien

Eine Klassifizierung von Warteschlangen kann erfolgen nach ihrer jeweiligen Bedienstrategie. Im Hinblick auf das im Kontext dieser Arbeit eingesetzte Werkzeug VITO sind insbesondere die folgenden Bedienstrategien relevant (vgl. [Müller-C., 2005], S. 12):

- **First-Come-First-Serve (FCFS)**[110], oft auch als First-In-First-Out (FIFO) bezeichnet: Eingehende Bedienaufträge werden in der Reihenfolge ihrer Ankunft abgearbeitet.
- **Processor Sharing (PS):** *N* Bedienaufträge teilen sich eine Verarbeitungseinheit, z. B. einen Prozessor. Ist nur ein Bedienauftrag anwesend, steht diesem die komplette Leistung des Prozessors zur Verfügung. Auf mehrere Aufträge entfällt die Leistung entsprechend zu gleichen Teilen ($^1/_N$).

[110] Zu den Bedienstrategien FCFS, PS und Alternativen vgl. [Kleinrock, 1976], S. 144ff. sowie [Lazowska, 1984], S. 128/129.

- **Infinite Server (IS)**[111]**:** Dieser Typ nimmt einen dedizierten Server für jeden Bedienwunsch an, wobei die Anzahl der Bedienwünsche unendlich groß werden kann, was typischerweise verwendet wird, etwa um die Denkzeiten von Anwendern zwischen einzelnen Arbeitsschritten zu modellieren.
- **Queue Dependent (QD):** Dieser Typ stellt einen Sonderfall des zuvor genannten PS dar, der verwendet werden kann, um Multiprozessor-Systeme abzubilden. In VITO wird ein solches System notiert als QD*n*, wobei *n* die Anzahl der Prozessoren beschreibt. Der Typ erlaubt es, das typische Verhalten von Multiprozessor-Systemen abzubilden, dass mit steigender Auslastung die Prozessoren nicht linear skalieren, sondern es zu Verlusten durch System-Overhead kommt. Pro Prozessor können hierfür Faktoren hinterlegt werden, so dass beispielsweise ein Dual-Prozessor bei Auslastung beider Prozessoren nicht den doppelten Durchsatz bewältigt, sondern nur um Faktor 1,9 schneller als ein einzelner Prozessor ist.

Eine Warteschlange kann durch Zufallsvariablen beschrieben werden, von denen einige Werte, mit denen das Modell parametrisiert wird, vorgegeben sind, während andere auf dieser Basis als gesuchte Performance-Werte berechnet werden können. So erreichen neue Bedienwünsche das System mit einem Ankunftsabstand A und es bezeichnet N die Anzahl an Aufträgen im Gesamtsystem, während N_W die Anzahl der derzeit in der Warteschlange befindlichen Aufträge und N_B die Anzahl der Aufträge in Bearbeitung ist. Die gesamte Verweilzeit (im Englischen Response Time) R eines Bedienwunsches im System ergibt sich als Summe der Wartezeit W und der Bedienzeit B ($R = W + B$). Als Parameter bei der Modellierung werden weiterhin die Ankunftsrate λ und Bedienrate μ verwendet, welche als Kehrwert von mittlerem Ankunftsabstand und mittlerer Bedienzeit definiert sind, sowie die Auslastung des Systems ρ als Quotient aus λ und μ.

Formel 11-4

$$\lambda = \frac{1}{E[A]}$$

Formel 11-5

$$\mu = \frac{1}{E[B]}$$

Formel 11-6

$$\rho = \frac{\lambda}{\mu}$$

[111] Zu den *Infinite* oder *Responsive Servers* vgl. [Kleinrock, 1975], S. 101ff.

Unabhängig davon, ob Ankunfts- und Bedienraten deterministisch sind oder nicht, gilt für ein solches Modell das Theorem von Little, im Englischen als „*Little's Law*" oder auch „*Little's Result*" bezeichnet, demzufolge die mittlere Anzahl von Bedienwünschen im System der mittleren Ankunftsrate multipliziert mit der mittleren Verweilzeit entspricht (vgl. [Kleinrock, 1975], S. 17).

Formel 11-7

$$E[N] = \lambda \cdot E[V] = \frac{E[R]}{E[A]}$$

Voraussetzung hierfür ist lediglich die Stationarität des Modells, d. h. es ist von einer langfristigen Betrachtung auszugehen, bei der die Bilanz von eingehenden und fertig bearbeiteten Bedienwünschen ausgeglichen ist und innerhalb der Warteschlange keine Bedienwünsche verloren gehen oder zusätzliche erzeugt werden. Zu weiteren Gesetzmäßigkeiten, die auf eine Warteschlange zutreffen siehe [Lazowska, 1984], S. 40ff. sowie [Menascé, 2004], S. 64ff.

11.2.3 Weitere Klassifikation – M/M/1

Eine weitere Klassifikation von Wartschlangen erfolgt gemäß der sogenannten *Kendall-Notation* nach dem Schema *A/B/c/K/m/Z* mit:

- *A* Verteilung der Ankunftszeiten
- *B* Verteilung der Bedienzeiten
- *c* Anzahl der Bedieneinheiten
- *K* Kapazität des Warteraums (inkl. der in Bedienung befindlichen Kunden)
- *m* Auftragsanzahl in der Quelle (spielt meist keine Rolle)
- *Z* Bedienstrategie

Oftmals wird jedoch nur die Kurzform der Kendall-Notation gemäß dem Schema *A/B/c* verwendet, wobei an die Stellen für *A* und *B* jeweils ein Bezeichner der stochastischen Verteilungsfunktion tritt, der Ankunfts- und Bedienzeit jeweils unterliegen. So bezeichnet *D* eine deterministische „Verteilung", also ein System, das nicht zufallsverteilt ist, sondern einer Konstante unterliegt. Bezogen auf IT-Systemtechnik würde etwa eine Komponente, die streng getaktet Transaktionen mit fixer Länge bearbeitet, als *D/D/1* notiert. *G* steht dagegen für eine arbiträre, nicht näher spezifizierte Verteilung, während *M* eine Exponentialverteilung bezeichnet. Der Bezeichner *M* steht dabei für die Markov-Eigenschaft der Exponentialverteilung, also ihre Gedächtnislosigkeit (vgl. Kapitel 11.1). Entsprechend wird ein System mit einem Prozessor und exponential verteilten Ankunfts- und Bedienzeiten als *M/M/1* notiert (vgl. [Kleinrock, 1975], S. 94ff.), ein Multi-Prozessor-System mit *c* gleichwertigen Prozessoren analog als *M/M/c*.

Eine solche Warteschlange und folglich auch ein Netz aus mehreren solcher Warteschlangen sind somit den analytischen Methoden aus dem Bereich der Markov-Ketten zugänglich, so dass für die stationäre Betrachtung relevante Kenngrößen und Leistungswerte einfach ermittelt

werden können. Es ergibt sich für $\rho < 1$ die Wahrscheinlichkeit P_0, die Warteschlange zu einem beliebigen Zeitpunkt leer vorzufinden, als

Formel 11-8

$$P_0 = 1 - \rho$$

und es folgt die Wahrscheinlichkeit P_k, zu einem Zeitpunkt k Kunden bzw. Bedienwünsche anzutreffen, als

Formel 11-9

$$P_k = (1 - \rho) \cdot \rho^k$$

Die mittlere Anzahl von Bedienwünschen $E[N]$, die sich im Warteschlangensystem befinden, kann daraufhin hergeleitet werden als

Formel 11-10

$$E[N] = \frac{\rho}{1 - \rho}$$

und unter Zuhilfenahme von Little's Law folgen die mittlere Verweilzeit $E[R]$, die mittlere Anzahl der wartenden Bedienwünsche $E[N_W]$ sowie die mittlere Wartezeit $E[W]$ als

Formel 11-11

$$E[R] = E[N] \cdot \frac{1}{\lambda} = \frac{\rho}{(1 - \rho) \cdot \lambda} = \frac{\lambda}{(1 - \rho) \cdot \lambda \cdot \mu} = \frac{1}{\left(1 - {}^{\lambda}/_{\mu}\right) \cdot \mu} = \frac{1}{\mu - \lambda}$$

Formel 11-12

$$E[N_W] = E[N] \cdot \rho = \frac{\rho^2}{1 - \rho}$$

Formel 11-13

$$E[W] = E[R] - E[B] = \frac{1}{\mu - \lambda} - \frac{1}{\mu} = \frac{\mu - (\mu - \lambda)}{(\mu - \lambda) \cdot \mu} = \frac{\lambda}{(\mu - \lambda) \cdot \mu} = \frac{\rho}{\mu - \lambda}$$

und weiter

$$\frac{\rho}{\mu - \lambda} = \frac{\rho \cdot \rho}{(\mu - \lambda) \cdot \rho} = \frac{\rho \cdot \rho}{\rho \cdot \mu - \rho \cdot \lambda} = \frac{\rho \cdot \rho}{\lambda - \rho \cdot \lambda} = \frac{\rho}{1 - \rho} \cdot \frac{\rho}{\lambda} = \frac{\rho}{1 - \rho} \cdot \frac{1}{\mu} = \frac{\rho \cdot E[B]}{1 - \rho}$$

Zu den detaillierten Hintergründen und Herleitungen der zuvorgenannten Zusammenhänge siehe „Birth-Death Processes" in [Kleinrock, 1975], S. 53[112] sowie „Generalized Birth-Death Models" in [Menascé, 2004], S. 279/280.

[112] Beim sogenannten Geburts- und Todes-Prozess (im Englischen „*Birth-Death Process*") handelt es sich um eine Sonderform einer Markov-Kette, bei der Zustandsübergänge ausschließlich zwischen direkt benachbarten Knoten stattfinden (vgl. [Kleinrock, 1975], S. 22).

Indem nun mehrere Stationen vom Typ M/M/1 verbunden werden, können Warteschlangennetze gebildet werden, deren Knoten die einzelnen Stationen bilden, während die Verbindungen zwischen den Knoten als gerichtete Graphen verstanden werden können, auf denen Bedienwünsche ihren Weg durch das Netz nehmen. Ein Nachteil dieses Ansatzes ist allerdings das in Kapitel 11.1 beschriebene Wachstum des Zustandsraums, wodurch komplexere Modelle mit zahlreichen Zuständen und Klassen nicht mehr effizient lösbar sind. Ein Ausweg besteht darin, statt einer Lösung des linearen Gleichungssystems einen rekursiven Ansatz zu nutzen. Hierzu kann der Algorithmus der Mean Value Analysis (MVA) verwendet werden.

11.2.4 Mean Value Analysis (MVA)

Der grundlegende MVA-Algorithmus basiert auf der Annahme einer einzigen geschlossenen Klasse von Kunden mit identischen Bedienwünschen, die exponential verteilt sind (vgl. [Lazowska, 1984], S. 114ff.). Weiterhin ist auch die Lösung von Modellen mit mehreren geschlossenen Klassen mittels MVA möglich (vgl. [Lazowska, 1984], S. 140ff.). Grundlage ist eine alternative Herangehensweise bei der Untersuchung der Zusammenhänge zwischen den einzelnen Performance-Werten des Modells.

Soll beispielsweise das in Abbildung 11-1 dargestellte Szenario um einen weiteren Kunden erweitert werden, so entspricht für den dritten Kunden die Wahrscheinlickeit, an einem der Knoten einen oder beide der ursprünglichen zwei Kunden vorzufinden, genau der Wahrscheinlichkeit, mit der sich im ursprünglichen Modell mit nur zwei Kunden einer oder beide Kunden in eben diesem Knoten befinden. Daraus folgt, dass die für den neuen Kunden zu erwartende mittlere Verweilzeit $R_k(n)$ an diesem Knoten verallgemeinert für ein Modell mit n Kunden sich aus der ursprünglichen Bedienzeit $B_k(n-1)$ im Modell mit $n-1$ Kunden bestimmen lässt

Formel 11-14

$$R_k(n) = B_k(n-1) \cdot [1 + \bar{n}_k(n-1)]$$

wobei der Term $\bar{n}_k(n-1)$ die mittlere Anzahl anwesender Kunden in Knoten k im ursprünglichen Modell mit $n-1$ Kunden bezeichnet. Bezeichnet V_k die Anzahl der Besuche von Knoten k pro Bedienwunsch, so folgt die Bedienzeit $B_0(n)$ des gesamten Warteschlangenneztes als Summe der Bedienzeiten aller Knoten K multipliziert mit der Besuchshäufigkeit je Knoten:

Formel 11-15

$$R_0(n) = \sum_{k=1}^{K} \acute{R}_k(n) = \sum_{k=1}^{K} [V_k \times R_k(n)]$$

Unter Zuhilfenahme von *Little's Law*, demzufolge sich die mittlere Anzahl von Kunden n in der Wartschlange aus dem Durchsatz $X_0(n)$ multipliziert mit der mittleren Bedienzeit $B_0(n)$ ergibt, folgt

Formel 11-16

$$X_0(n) = \frac{n}{R_0(n)}$$

Das sogenannte *Forced Flow Law* (vgl. [Lazowska, 1984], S. 47ff.) besagt ferner, dass der Durchsatz (X_k) eines einzelnen Knoten k dem Produkt aus mittlerer Anzahl an Besuchen pro Bedienwunsch (V_k) und dem Durchsatz des Gesamtsystems X_0 entspricht. Entsprechend ergeben sich die einzelnen Durchsätze eines jeden Knoten als

Formel 11-17

$$X_k(n) = V_k \times X_0(n)$$

Weiterhin ergibt sich gemäß dem sogenannten *Utilization Law* (vgl. [Lazowska, 1984], S. 41/42) die mittlere Auslastung eines einzelnen Knoten als Produkt aus dessen mittlerer Bedienzeit und dem mittleren Durchsatz

Formel 11-18

$$\rho_k(n) = B_k \times X_k(n)$$

Unter erneuter Zuhilfenahme von *Little's Law* folgt die mittlere Anzahl $\bar{n}_k(n)$ von Kunden in einem jedem Knoten k für ein Modell mit n Kunden als

Formel 11-19

$$\bar{n}_k(n) = X_k(n) \times R_k(n)$$

Mit der zuvor genannten Beziehung aus Formel 11-17 folgt

Formel 11-20

$$\bar{n}_k(n) = V_k \times X_0(n) \times R_k(n) = X_0(n) \times \acute{R}_k(n)$$

Die zuvor genannten Formeln beschreiben das iterative Vorgehen des MVA-Algorithmus. Basierend auf der mittleren Anzahl von Kunden pro Knoten in einem System mit $n-1$ Kunden werden die mittlere Bedienzeit pro Knoten für ein System mit n Kunden, darauf aufbauend die Bedienzeit des Gesamtsystems und schließlich der Durchsatz des Gesamtsystems und der einzelnen Knoten berechnet. Dies führt zur Auslastung der einzelnen Knoten sowie zur mittleren Anzahl von Kunden in jedem Knoten, woraufhin dann im nächsten Iterationsschritt der Algorithmus für $n+1$ Kunden durchlaufen werden kann. Initialisiert wird der Algorithmus mit der Annahme, dass $n = 0$ Kunden im System sind, was naturgemäß auch zu einer mittleren Anzahl von jeweils $\bar{n}_k(0) = 0$ Kunden in jedem Knoten führt. Der komplette MVA-Algorithmus lautet entsprechend ausformuliert:

Tabelle 11-1: Mean Value Analysis

Intialisiere $\bar{n}_k(0) = 0$ für alle Knoten k.

Berechne für jede Anzahl von Kunden $n = 1, 2, 3, \dots N$:

die mittlere Gesamtbedienzeit pro Knoten k:

$$\acute{R}_k(n) = V_k \times B_k(n) = V_k \times B_k(n-1) \cdot [1 + \bar{n}_k(n-1)]$$

die Bedienzeit des gesamten Systems:

$$R_0(n) = \sum_{k=1}^{K} \acute{R}_k(n) = \sum_{k=1}^{K} [V_k \times R_k(n)]$$

den Durchsatz des gesamten Systems:

$$X_0(n) = \frac{n}{R_0(n)}$$

den Durchsatz für jeden Knoten k:

$$X_k(n) = V_k \times X_0(n)$$

die Auslastung für jeden Knoten k:

$$\rho_k(n) = B_k \times X_k(n)$$

die mittlere Anzahl von Kunden für jeden Knoten k:

$$\bar{n}_k(n) = V_k \times X_0(n) \times R_k(n) = X_0(n) \times \acute{R}_k(n)$$

Quelle: Darstellung nach [Menascé, 2004], S. 323

Besteht der Vorteil des MVA-Algorithmus darin, dass damit eine Lösung auf diesem Weg wesentlich einfacher gefunden werden kann als durch die Lösung des linearen Gleichungssystems, so ist der Nachteil, dass die zuvor dargestellte Grundform des Algorithmus nur effizient für Warteschlangennetze mit wenigen geschlossenen Klassen von Kunden (=Lastketten) sowie lastunabhängigen Stationen funktioniert. Mehrere Erweiterungen des Algorithmus wurden daher entwickelt, um diese Anwendungsfälle zu adressieren.

Der Bedarf für mehrere Klassen ist insbesondere für die Entwicklung von Modellen im Bereich der IT-Systemtechnik gegeben, wo die Unterscheidung von mindestens zwei bis drei Klassen typischer Benutzer etwa nach *Light*, *Medium* und *Heavy User* (oder auch *Power User*) (vgl. [Köchling, 2004] und [Hiebel, 2011]) oder von mehreren Anwendungsprofilen (*Simple*, *Medium* oder *Complex Applications*, vgl. Kapitel 6.5.4) gebräuchlich ist.

Während der MVA-Algorithmus mit steigender Zahl von Kunden N und Stationen K in einem Wartschlangennetz linear skaliert, nimmt der Rechenaufwand mit steigender Anzahl an Klassen von Kunden exponentiell zu.

Sei eine Anzahl von Klassen R gegeben, so kann die Anzahl von Kunden in einem System als Vektor $\vec{N} = (N_1, N_2, N_3, \dots N_r)$ verstanden werden, wobei N_r die Anzahl von Kunden der Klasse r im System bezeichnet. Es folgt die Anzahl der nötigen Rechenoperationen (Multiplikationen

und Additionen), um für ein solches Modell die Lösung mittels des numerisch exakten MVA-Algorithmus (vgl. [Menascé], S. 353) zu ermitteln, als

Formel 11-21

$$KR\prod_{1}^{R}(1+N_r)$$

Hintergrund ist, dass die in Formel 11-14 benötigte mittlere Anzahl anwesender Kunden $\bar{n}_k$ in Knoten k für ein Modell mit $n-1$ Kunden nicht mehr nur für eine Klasse von Kunden berücksichtigt werden muss, sondern für alle R Klassen und somit für alle möglichen Ausprägungen des Vektors $\vec{N}$.

Die Berechnung der Werte mittels des MVA-Algorithmus erfordert entsprechend für jede konkrete Ausprägung die Berechnungen von R anderen Ausprägungen als Eingabe, wobei sich die mittlere Länge der Warteschlange $\bar{n}_{k,r}^{A}$ an einem Knoten k bei der Ankunft A eines Kunden der Klasse r ergibt als

Formel 11-22

$$\bar{n}_{k,r}^{A}(\vec{N}) = \bar{n}_{k,r}(\vec{N}-\vec{1}_r)$$

Um den notwendigen Rechenaufwand zu reduzieren, wurden an Stelle der exakten Methode des MVA-Algorithmus verschiedene angenäherte Varianten entwickelt, die schneller und dennoch mit akzeptabler Genauigkeit lösbar sind. So hat Yonathan Bard mit dem Large Customer Population (LCP) Algorithmus eine Approximierung erarbeitet (vgl. [Bard, 1979]).

Sein Ansatz basiert auf der Annahme, dass sich in Warteschlangennetzen mit einer sehr großen Anzahl an Kunden durch eine Verringerung der Anzahl um nur einen Kunden keine signifikante Änderung der mittleren Länge der Warteschlangen an den einzelnen Knoten ergibt. Für die Bestimmung von $\bar{n}_{k,r}^{A}$ folgt entsprechend

Formel 11-23

$\bar{n}_{k,r}(\vec{N}-\vec{1}_r) \approx \bar{n}_{k,r}(\vec{N})$ für jede Klasse r und somit

$$\bar{n}_{k,r}^{A}(\vec{N}) \approx \bar{n}_{k,r}(\vec{N})$$

Allerdings liefert der LCP Algorithmus für Netze mit einer geringen Anzahl an Kunden keine genauen Lösungen, weshalb Paul J. Schweitzer darauf aufbauend mit dem sogenannten Proportional Estimation (PE) Algorithmus (vgl. [Schweitzer, 1979]) einen approximierten Ansatz entworfen hat, der sowohl für kleine als auch für große Anzahlen von Kunden hinreichend genaue Ergebnisse liefert.

Der PE Algorithmus basiert auf der Annahme, dass sich die Anzahl von Kunden der Klasse r in einem jeden Knoten proportional verhält zur Anzahl aller Kunden der Klasse r im gesamten Warteschlangennetz. Es folgt daraus

Formel 11-24

$$\frac{\bar{n}_{k,r}(\vec{N}-\vec{1}_r)}{\bar{n}_{k,r}(\vec{N})} \approx \frac{N_r-1}{N_r} \text{ somit}$$

$\bar{n}_{k,r}(\vec{N}-\vec{1}_r) \approx \frac{N_r-1}{N_r}\bar{n}_{k,r}(\vec{N})$ und für die Verwendung im Algorithmus entsprechend

$$\bar{n}^A_{k,r}(\vec{N}) = \bar{n}_{k,r}(\vec{N}-\vec{1}_t) \approx \begin{cases} \bar{n}_{k,r}(\vec{N}) & t \neq r \\ \frac{N_r-1}{N_r}\bar{n}_{k,r}(\vec{N}) & t = r \end{cases}$$

bei einer Iteration der Parameter r und t jeweils über 1 bis R. Der Algorithmus beginnt mit seiner Iteration bei einer geschätzten Verteilung – aufbauend auf der Annahme, dass alle Kunden der Klasse r gleichmäßig über die Knoten des Netzes verteilt sind. Die Iteration endet, sobald sich $\bar{n}_{k,r}(\vec{N})$, basierend auf einer im Vorhinein festzulegenden Toleranzgrenze, dieser angenommenen Verteilung hinreichend genau angenähert hat (vgl. [Menascé], S. 359).

Den nach Bard/Schweitzer approximierten MVA-Algorithmus (vgl. [Menascé], S. 362) hat G. Totzauer ergänzt um ein Verfahren, das es erlaubt, Durchsatzsteuerung und -kopplung zu realisieren (vgl. [Totzauer, 1987]). Bei ersterem Ansatz handelt es sich um eine Zwangssteuerung, mit der der Durchsatz einer Lastkette fest eingestellt werden kann. Hierbei ist es allerdings möglich, dass der vorgegebene Durchsatz durch das betrachtete Warteschlangennetz nicht realisierbar ist. Der Algorithmus prüft dies entsprechend und löst das Netz nur für vorgegebene Werte, die durch das Gesamtsystem zu leisten sind.

Die Durchsatzkopplung erlaubt es, mehrere Lastketten zueinander in Relation zu setzen. D. h. es kann vorgegeben werden, dass eine Lastkette von einer anderen Lastkette abhängt (vgl. Kapitel 6.4.2 und 6.5.4). Soll eine Kette kk zu einer Kette k in Bezug gesetzt werden, so wird ihre Relation durch den Quotienten ihrer Durchsätze $D(kk)/D(k)$ ausgedrückt und als Koeffizient $koeff(kk)$ bezeichnet (vgl. [Totzauer, 1987], S. 365). Die Beziehungen der Ketten untereinander dürfen dabei allerdings nicht transitiv sein. Eine Kette, von der eine andere Kette abhängt, darf ihrerseits nicht wiederum von einer dritten Lastkette abhängig sein.

Der ursprüngliche Algorithmus nach Bard/Schweitzer wird dazu in der Weise ergänzt, dass eingangs alle Durchsätze bestimmt und daraufhin derart angepasst werden, dass die durch die Koeffizienten der Lastketten definierten Relationen abgebildet werden. Es ist dabei jedoch nur gestattet, Durchsätze von Lastketten zu verringern, was für darauf aufbauende Modelle bedeutet, dass die langsamste Lastkette die maximal möglichen Durchsätze aller anderen Lastketten im Warteschlangennetz vorgibt.

Weiterhin gilt, dass auch im Fall der Durchsatzkopplung zu prüfen ist, ob eine Lösung realisierbar ist. Sollte dies nicht der Fall sein, muss im Zweifelsfall auch die langsamste Lastkette in ihrem Durchsatz weiter reduziert werden, bis eine Lösung möglich ist. Der erweiterte

Algorithmus liefert als Lösung also die maximalen Durchsätze des gesamten Warteschlangennetzes unter Wahrung der vorher festgelegten Verhältnisse der Lastketten zueinander.

Eine Implementierung des Bard-/Schweitzer-Algorithmus mit den Erweiterungen von Totzauer zur Durchsatzkopplung und -steuerung[113] kommt in der Software VITO zum Einsatz (vgl. [Müller-C., 2001], S. III-8 sowie [Müller-C., 2005]. S. IV), wodurch die Anwendung geschlossene Warteschlangennetze auch mit vielen Knoten und mehreren, miteinander verknüpften Lastdefinitionen in sehr kurzer Zeit effizient lösen kann.

Die Software bildet entsprechend die Umgebung für die analytische Abbildung und Untersuchung von aus dem Referenzmodell für SBC Infrastrukturen abgeleiteten Informationsmodellen (vgl. Kapitel 5.4.2).

11.3 Clusteranalyse mittels k-Means-Algorithmus

Der sogenannte k-Means-Algorithmus ist eine Methode zur Gruppierung gleichartiger Objekte im Feld der Clusteranalyse. Ziel ist es dabei, die einzelnen Datensätze einer gegebenen Menge von Messwerten derart in Gruppen zusammenzufassen, dass die Varianz innerhalb der Gruppen im Vergleich zur Varianz aller Datensätze jeweils relativ klein ist (vgl. [Menascé], S. 106).

Der k-Means-Algorithmus unterteilt die zu untersuchende Datenbasis in k Cluster, wobei die gewünschte Anzahl k im Vorhinein festzulegen ist. Die Punkte eines jeden Clusters gruppieren sich um ihren Mittelpunkt, den sogenannten *Centroid*. Dieser wird aus dem Durchschnitt aller Punkte im Cluster ermittelt. Der Algorithmus beginnt mit der Auswahl von k Punkten innerhalb der gegebenen Menge, die als eine erste Näherung der Centroiden der zu bestimmenden k Cluster dienen.

Die übrigen Punkte werden dann schrittweise dem nächstgelegenen Centroiden zugeordnet, woraufhin die Centroiden neu berechnet und alle Punkte erneut zugeordnet werden, solange bis eine eine gegebene Anzahl von Durchläufen des Algorithmus ausgeführt wurde oder bis keine Punkte mehr ihren Cluster wechseln, d. h. bis ein stabiler Zustand erreicht ist. Das Vorgehen lautet entsprechend ausformuliert:

[113] Die Durchsatzsteuerung erfolgt durch die Verwendung einer Pseudostation, an der eine künstliche Verzögerung („Forced delay“) eingeführt wird, um das System gezielt langsamer laufen, als es maximal könnte.

Tabelle 11-2: k-Means-Algorithmus

1. Lege die Menge der zu ermittelnden Cluster k fest.
2. Bestimme die Koordinaten der k Centroiden als Ausgangspunkte. (Anfänglich kann es sich dabei um eine willkürliche Auswahl eindeutiger Punkte handeln oder etwa um die k Punkte der Menge, die am weitesten von einander entfernt liegen. Im weiteren Verlauf bilden dann die Mittelwerte aller Punkte im Cluster die neuen Centroiden).
3. Bestimme die Euklidische Distanz eines jeden Punktes der Menge zu allen Centroiden und ordne die Punkte ihren jeweils nächstgelegenen Centroiden zu. Berechne dann die Centroiden der Cluster neu, basierend auf den Mittelwerten aller Punkte im jeweiligen Cluster.
4. Wiederhole Schritt 3 solange, bis eine vorgegebene Anzahl von Durchläufen oder der stabile Zustand erreicht ist, bis also keine Punkte mehr in einen anderen Cluster wechseln.

Quelle: Darstellung nach [Menascé, 2004], S. 323

Bei der Anwendung des k-Means-Algorithmus in Kapitel 6.5.4 kommt eine frei verfügbare Implementierung des Entwicklers Sheldon Neilson zum Einsatz. Diese Implementierung basiert auf einem in einer Microsoft Excel Arbeitsmappe integrierten Visual BASIC for Applications (VBA) Makro[114].

[114] k-Means Cluster Analysis in Microsoft Excel: http://www.neilson.co.za/k-means-cluster-analysis-in-microsoft-excel/ (abgerufen am 01.02.2016).

12 Appendix C: Monitoring von SBC Infrastrukturen

Um eine Basis für die Validierung des in Kapitel 6 exemplarisch beschriebenen Einsatzes des Referenzmodells und seiner Methodik zu schaffen, wurden verschiedene Werkzeuge für das Monitoring der zugehörigen SBC Infrastruktur implementiert. Dieses Kapitel beschreibt, in welcher Weise Daten zur Nutzung und Auslastung der Umgebung erhoben wurden. Es stand dabei im Fokus, wie eine im produktiven Betrieb befindliche XenApp Terminal Server Farm im Hinblick auf ihre Nutzung und Auslastung überwacht werden kann.

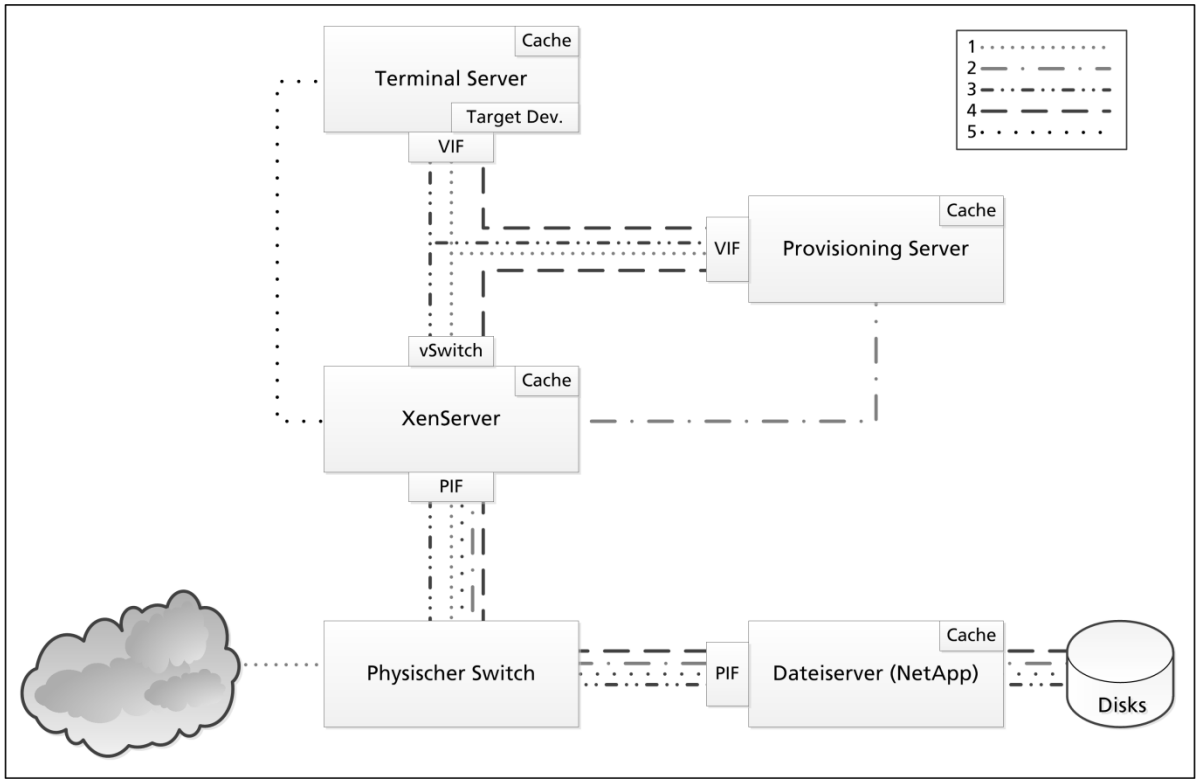
Zu berücksichtigende Aspekte waren zum einen das kurzfristige Monitoring mit dem Ziel, unmittelbar über Störungen, wie den Ausfall von einzelnen Diensten oder ganzen Servern, informiert zu werden und darauf reagieren zu können, und zum anderen die mittel- und langfristige Erfassung statistischer Daten. Letzteres steht in diesem Kapitel im Vordergrund und dient dem Kapazitätsmanagement mit dem Ziel, eine Basislinie zu gewinnen, welche den üblichen Betriebszustand der Umgebung widerspiegelt. Dies ermöglicht es, Abweichungen und Trends zu erkennen, um proaktiv die für die Anforderungen der Endanwender nötigen Ressourcen bereitstellen zu können und diese möglichst an höhere Anforderungen oder steigende Zahlen von Usern anzupassen, bevor es zu Engpässen kommt.

12.1 Zusammenhang von physischen und virtuellen Komponenten

Wenn es nun darum geht, das Zusammenspiel aller physischen und virtuellen Komponenten einer Umgebung zu untersuchen, sind vor allem Informationen darüber erforderlich, wie die Komponenten im Hinblick auf Netzwerk-, Storage und Prozessor-Nutzung interagieren. Besonders hinsichtlich der Input-/Output-Operationen pro Sekunde (IOPS) der – ebenfalls virtuellen und physischen – Festplatten ist innerhalb einer Virtualisierungsinfrastruktur genau zu betrachten, auf welcher Abstraktionsebene eine solche I/O-Operation stattfindet.

Dies stellt Abbildung 12-1 für verschiedene Anwendungsfälle dar. Unabhängig davon, ob eine VM provisioniert ist oder nicht, interagiert sie mit Ressourcen im die Infrastruktur umgebenden Netz über ihr virtuelles Netzwerk-Interface (VIF) sowie den ebenfalls virtuellen Switch (vSwitch) des XenServer (**1**). Dieser leitet die Kommunikation über sein physisches Netzwerk-Interface (PIF) an die physischen Netzwerkkomponenten, wie Switche, Router und Firewalls weiter – letztere werden in der Abbildung im Symbol der Wolke zusammengefasst. Dies trifft zu für sämtliche Anwendungen, welche Ressourcen außerhalb der betrachteten Infrastruktur in Anspruch nehmen, also beispielsweise das Drucken auf einem Print-Server oder das Abrufen von Webseiten. An letzterem Beispiel wird jedoch auch ersichtlich, dass nicht jede I/O-Operation auf Anwendungsebene zwingend im Netzwerk sichtbar wird, denn die Anforderung einer Webseite könnte z. B. lokal aus dem Cache der VM bedient werden, sofern die gewünschten Inhalte dort bereits vorhanden sind.

Abbildung 12-1: Zusammenhang physischer und virtueller Komponenten



Quelle: Eigene Darstellung

Bei einem Datei-basierten Zugriff auf die lokale Festplatte einer nicht provisionierten VM werden die Anforderungen vom XenServer umgesetzt (**2**) – im vorliegenden Beispielszenario auf die der virtuellen Festplatte entsprechende Datei, die per Network File System (NFS) vom Dateiserver verbunden ist. Auch in diesem Fall schlägt sich eine I/O-Operation aber nur im Netzwerk nieder, wenn sie nicht bereits aus dem lokalen Cache der jeweiligen VM bedient werden kann.

Wiederum unabhängig von der konkreten Ausprägung einer VM – provisioniert oder nicht – werden Datei-Operationen, die nicht auf der lokalen Festplatte der VM arbeiten, sondern per Common Internet File System (CIFS) auf dem NetApp Dateiserver (**3**) über das VIF, den vSwitch und die physische Netzwerk-Infrastruktur geleitet, soweit sie nicht aus dem Cache der VM bedient werden können. Der NetApp Dateiserver verfügt seinerseits ebenfalls über einen Cache und bedient lesende Anfragen ggf. zunächst aus diesem bzw. speichert Schreibzugriffe zur späteren Ausführung zwischen.

Anders verhält es sich mit I/O-Operationen auf der lokalen Festplatte einer provisionierten VM, da diese aus einem schreibgeschützten „Golden Image“ erzeugt wird (vgl. Kapitel 2.3.1). Lesende Zugriffe werden über den Target Device Treiber der Provisioning Services mittels eines proprietären Netzwerk-Protokolls über das VIF der virtuellen Maschine zum Provisioning Server geleitet (**4**), sofern es sich nicht um eine Datei handelt, die sich auf der Cache-Disk der

provisionierten VM befindet (siehe folgender Absatz). Der Provisioning Server bedient die Anfrage entweder aus seinem Cache oder kommuniziert per CIFS über sein VIF, den vSwitch und die physische Infrastruktur mit dem NetApp Dateiserver, der die Anfrage wiederum aus seinem Cache bedient oder letztendlich auf seine Festplatten zugreift.

Wenn die provisionierte VM eine bestehende Datei verändert oder neue Daten auf ihre lokale Festplatte schreibt, ist dies nicht unmittelbar möglich, da das zu Grunde liegende Image schreibgeschützt ist. Die VM verfügt daher über eine weitere lokale Festplatte, die wie im Fall einer nicht provisionierten VM als virtuelle Festplatte im Kontext des XenServers per Network File System (NFS) vom Dateiserver verbunden ist und als Cache des Target Device Treibers dient (**5**). Gegenüber dem „Golden Image" geänderte und neue Daten werden also auf diesem Pfad geschrieben und gelesen, so dass I/O-Operationen innerhalb einer provisionierten VM unterschiedliche Pfade nehmen – je nachdem, wo sich die betroffenen Daten befinden.

Aus diesen Zusammenhängen wird ersichtlich, dass I/O-Operationen auf Anwendungs- bzw. Betriebssystem-Ebene innerhalb der VM ihren Weg über mehrere Pfade sowie verschiedene Protokolle nehmen können und dabei ggf. auf mehrere Zwischenspeicher zurückgreifen. Eine eindeutige Zuordnung von IOPS innerhalb der VM und tatsächlichen IOPS auf den physischen Festplatten des NetApp Dateiservers ist somit nicht möglich und es kann die Last an mehreren Messpunkten innerhalb der Infrastruktur nur näherungsweise bestimmt werden – beispielsweise, indem über einen Beobachtungszeitraum Last und Benutzeranzahl innerhalb der VM gemessen und in Relation zur ebenfalls gemessenen Gesamtlast auf XenServer und NetApp gesetzt werden. Techniken zur Erhebung dieser Messdaten werden im Folgenden vorgestellt und dienen zur weiteren Parametrisierung des in Kapitel 6.4 entworfenen VITO-Modells.

Monitoring und Langzeitstatistik der Umgebung erfolgen maßgeblich über die Systeme LANrunner sowie Citrix EdgeSight. Die EdgeSight-Agenten sind als Software-Komponente direkt auf den Terminal Servern installiert. Die erforderlichen Server-Komponenten werden dagegen außerhalb des zu untersuchenden Systems betrieben, um die zusätzliche Last durch das Monitoring so gering wie möglich zu halten und die Ergebnisse nicht zu verfälschen.

12.2 Netzwerk-Traffic und allgemeine Leistungsdaten – LANrunner

Zur Erhebung von Datenübertragungsraten im Netzwerk sowie allgemeinen Leistungsdaten der zu untersuchenden Systeme stand mit dem LANrunner[115] ein von Fraunhofer UMSICHT selbst

[115] „Mit Hilfe von LANrunner® lassen sich Netzwerkstatistiken ermitteln und visualisieren, so dass IT-Verantwortliche und Administratoren jederzeit die Übersicht behalten und wichtige Erkenntnisse für ihre Planung gewinnen. Die Messwerterfassung mittels SNMP erlaubt dabei neben der Ermittlung von Volumen die Erfassung aller per SNMP abfragbaren numerischen Größen, wie z. B. die Arbeitsspeicher- und Festplattenauslastung oder auch die Prozessortemperatur eines Servers. Eine intuitiv zu bedienende Weboberfläche auf Basis aktueller Web-Technologien sowie die offene Schnittstelle zur Datenbank erlauben die flexible Verwendung und Visualisierung der Daten. Die Werte stehen über einen Zeitraum von Monaten und sogar Jahren zur Auswertung bereit. Neben der Identifizierung nicht bestimmungsgemäßer Zustände liefert die Langzeitstatistik ein wertvolles Planungsinstrument, so dass Ressourcen frühzeitig und begründet erweitert werden können." (Quelle: Fraunhofer Competence Center LAN)

entwickeltes Werkzeug zur Verfügung, welches via Simple Network Management Protocol (SNMP)[116] beliebige Werte im Netzwerk abfragen kann und es so ermöglicht, umfassende statistische Daten zur Auslastung von Windows Servern zu erfassen. Dies beinhaltet allgemeine Daten, die auch über die Windows-eigene Leistungsüberwachung – unter Windows Server 2003 noch als „Systemmonitor" bezeichnet – verfügbar sind, unter anderem Daten zur Auslastung von Prozessoren, Hauptspeicher, Festplatten und Netzwerkkarten.

12.2.1 Netzwerkkarten

Den auf der Netzwerkkarte eines Windows Servers eingehenden Traffic liefert der folgende OID vom Typ „Counter32":

```
.1.3.6.1.2.1.2.2.1.10.12
```

Den von der Netzwerkkarte ausgehenden Traffic liefert der folgende OID vom Typ „Counter32":

```
.1.3.6.1.2.1.2.2.1.16.12
```

Je nach Ausstattung eines Systems ist ggf. mit dem Kommandozeilenbefehl „snmpwalk" oder dem Tool „Getif" zu prüfen, welche Netzwerkkarten auf einem Zielsystem existieren. Analog können mittels LANrunner sämtliche an der Netzwerkkommunikation der Server beteiligten Komponenten, wie Switches, Firewalls, Router oder Hypervisor-Hosts, überwacht werden. Welche OID ein Gerät für Abfragen bereitstellt, kann im LANrunner über einen MIB Browser ermittelt werden, der per „snmpwalk" sämtliche Informationen auflistet und mittels der vom jeweiligen Hersteller bereitgestellten MIB-Dateien in lesbarer Form darstellt.

Wichtig ist im Hinblick auf die Auswertung der verschiedenen OID deren Datentyp. Der zuvor genannte „Counter", den es als 32- oder 64-Bit Variante gibt, besitzt einen Wertebereich der ganzen Zahlen von 0 bis $2^{32}-1$ bzw. $2^{64}-1$ und die Eigenschaft, dass der in einem „Counter" gespeicherte Wert über die Zeit monoton steigt und im regulären Betrieb niemals abnimmt. Wird das Maximum des Wertbereichs erreicht, springt die Variable auf 0 und beginnt erneut zu steigen (vgl. [Mauro, 2005], S. 26). Über das Delta zwischen zwei Abfragen können Rückschlüsse auf das Systemverhalten gezogen werden. Im hier beschriebenen Fall von Netzwerkkarten und -ports weist die Differenz der Werte einer „Counter"-Variablen zu zwei Mess-Zeitpunkten aus, wie viele Daten im Zeitraum zwischen den Abfragen übermittelt wurden. Der LANrunner ermittelt diese Deltas alle fünf Minuten, so dass der Graph bzw. die

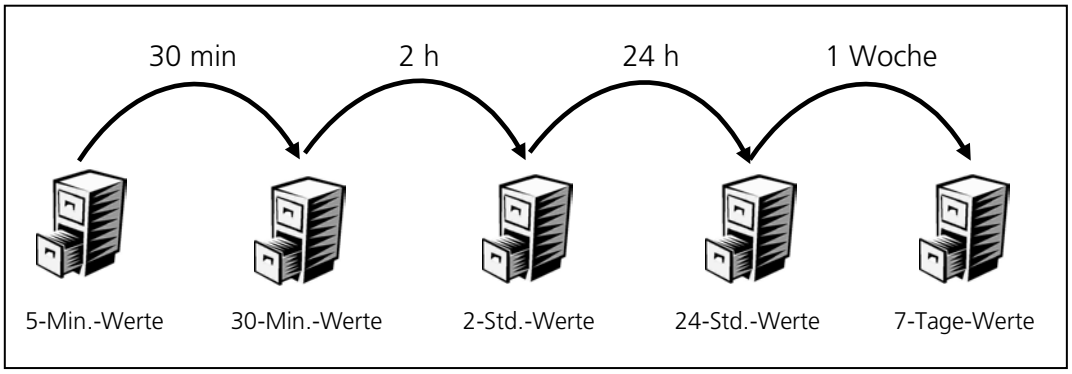
[116] Beim Simple Network Management Protocol (SNMP) handelt es sich um ein Protokoll auf dem Application Layer des ISO/OSI-Referenzmodells, welches auf UDP als Transport-Protokoll aufsetzt. Es erlaubt das Monitoring Netzwerk-fähiger Geräte, wie Switches, Router, Drucker sowie Server und Client-Computer. Welche Werte per SNMP abgefragt werden können, ist nicht fest im Protokoll veranktert. Stattdessen sind die Werte in einer hierarchischen Datenbankstruktur, der Management Information Base (MIB), organisiert, deren Schema von Gerät zu Gerät unterschiedlich sein kann (vgl. [Mauro, 2005], S. 23ff.). Einzelne Werte innerhalb der MIB werden als Object Identifier (OID) bezeichnet.

exportierten Werte für Fünf-Minuten-Intervalle gemittelte Werte der letzten zwei Tage ausweisen.

Darüber hinaus werden die gespeicherten Messwerte in verschiedenen Stufen konsolidiert, so dass die Größe der Datenbank auch über lange Zeit nahezu konstant bleibt. Die Werte liegen als durchschnittliche minimale und maximale Werte vor, so dass durch die Konsolidierung nur ein geringer Informationsverlust entsteht. Auf diese Weise ist es möglich, die Auslastung von Servern und Netzwerkkomponenten über einen Zeitraum von Monaten oder gar Jahren zu beobachten. Engpässe lassen sich auf diese Weise frühzeitig erkennen.

In Abbildung 12-2 werden die Konsolidierungsstufen vereinfacht dargestellt. Die alle fünf Minuten gemessenen Werte werden für zwei Tage, die 30-Minuten-Werte für 14 Tage, die Zwei-Stunden-Werte für 60 Tage, die 24-Stunden-Werte für zwei Jahre und die Wochenwerte für zehn Jahre vorgehalten.

Abbildung 12-2: Datenkonsolidierung im LANrunner



Quelle: Eigene Darstellung

Für den in Kapitel 6.5.1 ermittelten Zeitraum der Untersuchung wurde die Netzwerkauslastung der Clients, Server und Netzwerkkomponenten entsprechend aus dem LANrunner abgefragt.

12.2.2 Hauptspeicher

Den zugesicherten Speicher – physisch und virtuell – in Benutzung liefert der folgende OID vom Typ „Integer“:

```
.1.3.6.1.2.1.25.2.3.1.6.4
```

Der folgende OID liefert den freien physischen Hauptspeicher in MB als Wert vom Typ „Integer“:

```
.1.3.6.1.2.1.25.2.3.1.6.5
```

12.2.3 Festplatten

Nicht nur für Dateiserver ist ein kontinuierliches Monitoring des freien Speicherplatzes interessant. Bei Terminal Servern, die mittels Citrix Provisioning Services bereitgestellt werden, steht vor allem eine Überwachung der System- und Cache-Disks im Fokus. Die OID unterhalb des Trees .1.3.6.1.2.1.25.2.3.1.6 liefern auch hierzu entsprechende Werte. Je nach Anzahl der Festplatten bzw. Partitionen und deren Laufwerksbuchstaben variieren aber die OID von System zu System. Daher ist zunächst ein „snmpwalk" auf dem zu messenden System erforderlich, um die gewünschten OID zu ermitteln. Auf den untersuchten Terminal Servern fand sich der freie Speicherplatz der Partitionen „C:" und „E:" in Prozent ebenfalls als Wert vom Typ „Integer" z. B. unter folgenden OIDs:

```
.1.3.6.1.2.1.25.2.3.1.6.1
.1.3.6.1.2.1.25.2.3.1.6.3
```

12.2.4 Prozessor

Die Prozessorinstanzen eines Servers können über folgende OID vom Typ „Integer" abgefragt. werden, was auf den betrachteten virtuellen Terminal Servern mit jeweils vier vCPU möglich war über:

```
.1.3.6.1.2.1.25.3.3.1.2.4
.1.3.6.1.2.1.25.3.3.1.2.5
.1.3.6.1.2.1.25.3.3.1.2.6
.1.3.6.1.2.1.25.3.3.1.2.7
```

Je nach Ausstattung eines Systems ist ggf. mit dem Kommandozeilenbefehl „snmpwalk" oder dem Tool „Getif" zu prüfen, welche Prozessorinstanzen auf einem Zielsystem existieren.

Im Gegensatz zum Datentyp „Counter" können die Werte von OID des Typs „Integer" (32-Bit) über die Zeit sowohl zu- als auch abnehmen (vgl. [Mauro, 2005], S. 26). D. h. die Abfrage eines solchen OID repräsentiert immer nur eine Momentaufnahme zum Abfragezeitpunkt. Da auch diese Werte vom LANrunner in Intervallen von fünf Minuten abgefragt werden, muss die Aussagekraft insbesondere für stark volatile Komponenten kritisch hinterfragt werden. Mag die Überwachung des belegten und freien Festplattenplatzes auf diese Weise noch möglich sein, eignet sich die Überwachung mittels SNMP für die Prozessoren eines Servers kaum. Betrüge die Auslastung eines Prozessors zu einem Abfragezeitpunkt 25 Prozent und bei der nächsten Abfrage im Fünf-Minuten-Intervall 30 Prozent, so wäre keinerlei Aussage über die Entwicklung der Last *während* des Fünf-Minuten-Intervalls möglich. Steigt die Last etwa *zwischen* zwei Messungen kurzfristig auf über 90 Prozent, so würde dies nicht in die Statistik eingehen.

Es ist also ein anderes Herangehen im Hinblick auf die Untersuchung der Server erforderlich. Insbesondere für das Monitoring von Terminal Servern empfiehlt es sich zudem weiterhin, die Anzahl der aktiven Sessions in Verbindung mit den Werten zur CPU- und Speicherauslastung

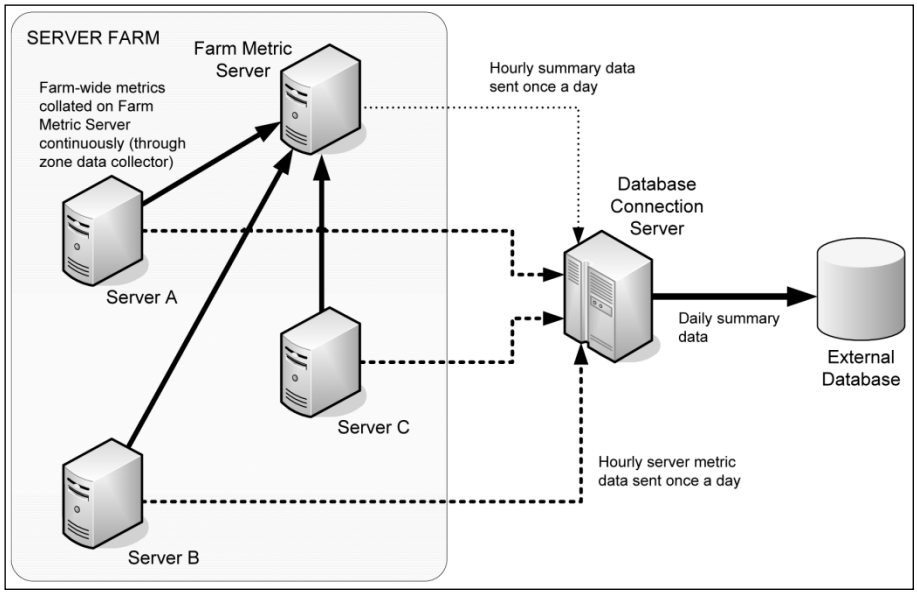
zu messen, um Erkenntnisse darüber zu gewinnen, wie gut die Hardware in Relation zur Anzahl der angemeldeten Benutzer skaliert, um frühzeitig durch Erweiterung der Farm um zusätzliche Terminal Server reagieren zu können.

Weitergehende Informationen, die besonders für die Administratoren von Terminal Servern von Interesse sind, lassen sich per SNMP jedoch leider nicht ermitteln. So ist es beispielsweise nicht möglich zu erfassen, wie oft eine bestimmte veröffentlichte Applikation über einen definierten Zeitraum aufgerufen wurde und welche Ressourcen diese Applikation genutzt hat. In früheren Versionen des Presentation Server bis hin zu XenApp 5.0 für den Windows Server 2003 hatte Citrix zu diesem Zweck eine Funktion namens „Resource Manager" integriert, die es erlaubt, Daten zur Nutzung von Applikationen zu erheben. Aus technischen Gründen, die im Folgenden erläutert werden, wurde der Resource Manager vom Hersteller aber nicht weiter gepflegt, sondern ab XenApp 5.0 durch Citrix EdgeSight ersetzt, welches in der Enterprise sowie der Platinum Edition von XenApp enthalten ist.

12.3 Vergleich von Resource Manager und EdgeSight

Das grundlegende Funktionsprinzip von Resource Manager und EdgeSight ähnelt sich sehr. Beide sammeln Daten über Agenten auf den Terminal Servern und übermitteln diese an eine relationale Datenbank auf Basis des Microsoft SQL Server. Beim Resource Manager handelt es sich um eine fest in den Presentation Server bzw. XenApp integrierte Komponente, welche entsprechend ausschließlich Terminal Server überwachen kann und dies auch nur innerhalb einer Farm. Die Konfiguration des Resource Manager erfolgt über die Java-basierende Konsole des Presentation Server und zur Authentifizierung ist entsprechend der Account eines Farm Administrators erforderlich.

Abbildung 12-3: Citrix Resource Manager Architektur



Quelle: [Citrix, 2006], S. 52

Bezogen auf die gesamte Farm werden übergreifende Informationen erfasst, wie z. B., welche von mehreren Servern veröffentlichte Anwendung insgesamt wie oft gestartet wurde, ebenso wie Farm-weite Ereignisse, so z. B. der Ausfall eines Servers. Pro Server werden zudem Leistungsdaten und Informationen zu den laufenden Sitzungen erfasst. Die Farm-weiten Informationen werden von einem Farm-Auswertungsserver gesammelt, stündlich konsolidiert und an den Datenbankverbindungsserver der Farm übermittelt. Die Server-spezifischen Daten werden lokal auf jedem Terminal Server erfasst, ebenfalls stündlich konsolidiert und an den Datenbankverbindungsserver übertragen. Dieser konsolidiert die Daten auf täglicher Basis und speichert sie in der externen Microsoft SQL Datenbank (vgl. Abbildung 12-3).

Da der Resource Manager jedoch ausschließlich auf den Presentation Server bzw. auf XenApp fokussiert und auch bezüglich der erfassten Daten und deren Auswertung beschränkt ist, wurde die Komponente zuletzt in XenApp 5.0 implementiert und nicht weiterentwickelt. Stattdessen hat Citrix mit der Übernahme des Software-Herstellers Reflectent Software im Jahr 2006 dessen Produkte in das eigene Portfolio integriert und fortan als Citrix EdgeSight angeboten. EdgeSight ist in mehreren Varianten verfügbar:

- **EdgeSight for XenApp/XenDesktop** bietet sowohl für Terminal Server als auch für virtuelle Desktops proaktives Echtzeit-Monitoring sowie die Erfassung statistischer Daten über einen längeren Zeitraum. Es werden dazu Daten zur allgemeinen Auslastung der Server/Desktops wie auch zur Performance und Ressourcen-Nutzung einzelner Benutzersitzungen erfasst. Die Lösung unterstützt damit die Trendanalyse und Kapazitätsplanung.

- **EdgeSight for Endpoints** bietet die Überwachung von veröffentlichten Applikationen bis hin zum Arbeitsplatz des Endanwenders. Beim Ende-zu-Ende Monitoring steht die Erfassung der vom Benutzer wahrgenommenen Arbeitsgeschwindigkeit im Vordergrund, d. h. die Frage, ob das Antwortzeitverhalten von via ICA bereitgestellten Applikationen akzeptabel ist und die Anwendungen in angemessener Zeit auf Tastatur- und Maus-Eingaben reagieren. Dazu wird unter anderem die Latenz und Bandbreite der Netzwerkverbindung überwacht, um Rückschlüsse auf die Performance des ICA-Protokolls zu ziehen. In Verbindung mit EdgeSight for XenApp kann bei vom Benutzer wahrgenommer langsamer Reaktion von veröffentlichten Anwendungen festgestellt werden, ob die Ursache auf dem Terminal Server selbst oder auf der Strecke zwischen Server und Client zu suchen ist.

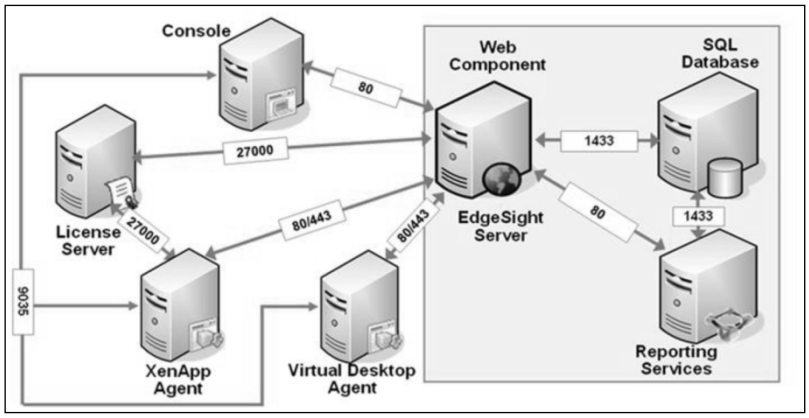
- **EdgeSight for Load Testing** ermöglicht automatisierte Lasttests, indem skriptbasiert Testbenutzer angemeldet werden und in deren Kontext definierbare Aufgaben durchgeführt werden. Dazu kann die Interaktion eines Benutzers mit dem System aufgezeichnet und ebenfalls skriptiert wieder abgespielt werden, um zahlreiche Benutzer zu simulieren und so Engpässe zu identifizieren, bevor ein System in den Produktivbetrieb überführt wird. Diese Variante von EdgeSight wird vom Hersteller

nicht mehr weiterentwickelt. Der Support wurde relativ kurzfristig zum Ende des Jahres 2013 abgekündigt[117], da Citrix nun den im Bereich von Terminal Servern und virtuellen Desktops zum „Quasi-Standard" avancierten Login Virtual Session Indexer (Login VSI)[118] als Lösung für automatisierte Performance-Tests empfiehlt.

- **EdgeSight for NetScaler** dient dem Monitoring von Web-Applikationen, die über den Citrix NetScaler[119] bereitgestellt werden und ermöglicht das proaktive Monitoring von Performance und Antwortzeitverhalten dieser Anwendungen.

Da in Hinsicht auf den Betrieb von Terminal Servern nur EdgeSight for XenApp den Windows Server 2008 R2 unterstützt, wird im weiteren Verlauf nur dieses weiter betrachtet und nicht die anderen EdgeSight Varianten oder gar der Resource Manager. Gegenüber letzterem bietet EdgeSight mehrere Vorteile. So ist EdgeSight im Gegensatz zum Resource Manager nicht fest in XenApp integriert, sondern wird als separater Agent installiert und somit als separater Prozess unabhängig von XenApp ausgeführt. Eine EdgeSight Server Installation ist damit nicht auf eine Terminal Server Farm beschränkt. Vielmehr ist die Lösung mandantenfähig und kann mehrere Terminal Server Farmen und sogar unterschiedliche Kunden verwalten, wobei pro Kunde ein separater Sicherheitsbereich mit eigener Konfiguration und eigenen Reports zur Auswertung der Daten verwaltet werden kann.

Abbildung 12-4: Citrix EdgeSight Architektur



Quelle: Citrix Systems, 2012

Zentrale Komponente des Systems ist der EdgeSight Server, der die Leistungsdaten von Clients, auf denen der EdgeSight Agent installiert ist, empfängt und in einer SQL Datenbank speichert.

[117] Lifecycle Milestones for EdgeSight: http://www.citrix.de/support/product-lifecycle/milestones/edgesight.html (abgerufen am 01.02.2016).

[118] Login VSI: http://www.loginvsi.com/product-overview (abgerufen am 01.02.2016) und https://support.citrix.com/article/CTX125630 (abgerufen am 01.02.2016).

[119] Beim NetScaler handelt es sich um eine Appliance, die dazu dient, die Bereitstellung von Web-Anwendungen und -Diensten zu sichern, zu optimieren und zu beschleunigen. Der NetScaler wird dazu Web- oder Datenbank-Servern vorgeschaltet und agiert als Load-Balancer, SSL-Beschleuniger und Cache sowie als Application Level Firewall.

Konfiguration und Auswertung erfolgen über die EdgeSight Console, die auf den Microsoft eigenen Webserver „Internet Information Server (IIS)“ aufsetzt und die Reporting Services des SQL Servers nutzt (vgl. Abbildung 12-4). Die Konsole kann mittels Internet Explorer von einem beliebigen Windows-Client aufgerufen werden. Einige Report-Funktionen erfordern darüber hinaus Microsoft Excel, den Adobe Flash Player und den Adobe Reader. Die Möglichkeiten zur Visualisierung und zum Export der Daten gehen damit deutlich über die Fähigkeiten des Resource Managers hinaus, der lediglich in der Lage ist, Berichte im HTML-Format zu erzeugen. EdgeSight bietet demgegenüber neben HTML auch Exporte als PDF, Excel-Tabelle bzw. kommaseparierte Liste oder XML.

12.4 EdgeSight-Installation und -Konfiguration

Der EdgeSight Server wurde als virtuelle Maschine auf einem separaten Hypervisor außerhalb der zu untersuchenden Infrastruktur installiert. Zuvor wurde mit dem „EdgeSight Database Size Estimation Tool“[120] der zu erwartende Speicherplatz-Bedarf für die EdgeSight Datenbank ermittelt.

Da im Vorfeld des Monitorings mit EdgeSight noch keine detaillierten Informationen zur Auslastung der Farm vorlagen (eben diese zu gewinnen, war ja das Ziel der EdgeSight Installation), wurde das Tool mit den von Citrix vorgegebenen Standard-Werten parametrisiert und es wurde lediglich die Anzahl der Server angepasst.

Entsprechend wurde ausgehend von den zum Zeitpunkt der Installation aktuell 17 Servern zunächst eine Gesamtgröße der Datenbank mitsamt Logfiles von fünf bis zehn GB beim Vorhalten der Performancedaten für 30 Tage kalkuliert (vgl. Abbildung 12-5). Sollen die Daten für 90 Tage gespeichert werden, erhöht sich der Speicherplatzbedarf auf 14 bis 25 GB.

Bei einem zukünftigen Wachstum der Server-Farm auf Grund von steigenden Benutzerzahlen sowie steigenden Anforderungen seitens der Anwender wurde zudem eine zukünftige Ausweitung der Farmen auf bis zu 30 Server kalkuliert, was in einer zu erwartenden Größe der Datenbank mit Logfiles von neun bis 16 GB für 30 Tage resultiert.

Bei 90 Tagen erhöht sich der Platzbedarf auf 23 bis 43 GB. Die VM wurde daraufhin zunächst mit einer Festplattenkapazität von 100 GB angelegt und es wurden vier virtuelle Prozessoren sowie acht GB Hauptspeicher zugewiesen.

[120] EdgeSight Database Size Estimation Tool: http://support.citrix.com/article/CTX122146 (abgerufen am 01.02.2016).

Abbildung 12-5: Ermittlung der zu erwartenden Datenbank-Größe

Citrix EdgeSight™ database sizing

Estimated Total Data Bytes	22.849.534.099	13.413.357.447
Estimated Total Data GBytes	21,28	**12,49**
Estimated total disk space needed*	**23 - 43 GB**	**14 - 25 GB**
	Future estimate	Current/Baseline estimate
Endpoints		
Number of End User Devices	0	**0**
Average number of Processes running	50,00	50,00
Average number of Sites Visited per User per day	40,00	40,00
Average number of Errors per User per day	5,00	5,00
Number of Hours in the Work Day for an endpoint	8,00	8,00
	Future estimate	Current/Baseline estimate
XenApp		
Number of XenApp Servers	**30**	**17**
Average number of System Processes running	50,00	50,00
Average number of Users per XenApp Server	40,00	40,00
Average number of running Published Apps per user	5,00	5,00
Average number of running Sessions per User	1,00	1,00
Average number of Sessions Started per User per hour	3,00	3,00
Average number of Sites visited per User per day	40,00	40,00
Average number of Errors per User per day	1,00	1,00
Number of Hours in the Work Day for a XenApp Server	24,00	24,00
Active Application Monitoring XenApp Servers	30	17
Active Application Monitoring Tests run per day per serve	0,00	0,00
Active Application Monitoring Points per test	1,00	1,00
	Future estimate	Current/Baseline estimate
Grooming Settings (days)		
System Performance	90	90
Process Performance	90	90
Process Usage	90	90
Network Performance	90	90
Network Transactions	90	90
Disk Usage	90	90
Alerts	90	90
XenApp System Performance	90	90
Session Performance	90	90
ICA Roundtrip Performance	90	90
XenApp Server Start Performance	90	90
ICA Client Start Performance	90	90
XenApp Environmental Usage	90	90
XenApp Application Response	90	90
ICA Channel Performance	90	90

* Database Transaction logs can be 10%-100% of the data size. The "Estimated total disk space needed" value is a combination of In-use DB storage and free space for logs. Customers can install SQL filegroups (CTX117433) onto different disks on the SQL server (thus spreading out the estimated size onto many disks).

Delta Data Bytes	9.436.176.653
Delta Data GB	8,79

Quelle: Eigene Abbildung nach Citrix, 2009

12.4.1 Vorbereitungen

Die VM wurde daraufhin unter dem Namen „edgesight“ mit dem Betriebssystem Windows Server 2008 R2 (SP1) eingerichtet und in die Windows Active Directory Domäne aufgenommen. Dann wurden die von EdgeSight benötigten Systemkomponenten und der Microsoft SQL Server 2008 R2 in der Standard Edition installiert. Dazu wurde zunächst im Server-Manager über die Aktion „Rollen hinzufügen“ der Webserver (IIS) mit allen geforderten Rollendiensten installiert:

- Statischer Inhalt
- Standarddokument
- ASP.NET
- ISAPI-Erweiterungen
- ISAPI-Filter
- Windows-Authentifizierung
- Anforderungsfilterung
- IIS 6-Verwaltungskompatibilität
    - IIS 6-Metabasiskompatibilität
    - IIS 6-WMI-Kompatibilität
    - IIS 6-Skriptingtools
    - IIS 6-Verwaltungskonsole

Weiterhin wurden zunächst über die Aktion „Features hinzufügen“ das .NET Framework 3.5.1 sowie der Message Queing Server und dann im Anschluss Microsoft Excel aus dem Funktionsumfang von Microsoft Office 2007 vollständig mit allen Office-Tools und der Standard-Auswahl an gemeinsam genutzten Office-Features installiert.

Zusätzlich wurden der Adobe Reader sowie der Adobe Flash Player als ActiveX Control für den Internet Explorer hinzugefügt. Daraufhin wurden das Betriebssystem und die Office-Installation mit allen von Microsoft angebotenen, wichtigen Sicherheitsupdates aktualisiert. Nach einem obligatorischen Neustart folgte die benutzerdefinierte Installation des SQL Server 2008 R2 mit folgenden Komponenten:

- Datenbankmoduldienste
    - SQL Server-Replikation
    - Volltextsuche
- Reporting Services
- Konnektivität der Clienttools
- Verwaltungstools – Einfach
    - Verwaltungstools – Vollständig

Der Dienst „SQL Server-Agent“ wurde im Sicherheitskontext von NT-AUTORITÄT\SYSTEM eingerichtet, die Dienste „SQL Server Database Engine“ und „SQL Server Reporting Services“ im Kontext von NT-AUTORITÄT\NETZWERKDIENST. Zur Anmeldung kommt ausschließlich der Windows-Authentifizierungsmodus zum Einsatz. Dabei wurde das Konto des Domänen-Administrators als SQL Server Administrator hinzugefügt. Anschließend wurde das Service Pack 2 für den SQL Server 2008 R2 installiert, gefolgt von einem weiteren Neustart. Nun wurden die Sicherheitseinstellungen der Reporting Services angepasst, indem lokal auf dem EdgeSight Server der URL „http://edgesight/Reports“ aufgerufen wurde. Dann wurde innerhalb der „Ordnereinstellungen“ des Ordners „Stamm“ die Aktion „Bearbeiten“ für die Gruppe „VORDEFINIERT\Administratoren“ ausgeführt und es wurden alle verfügbaren Rollen zugewiesen.

12.4.2 EdgeSight-Server und -Datenbank

Damit waren die Vorbereitungen abgeschlossen und es folgte die Installation der Citrix EdgeSight Server-Website und -Datenbank in der Version 5.4.6.3. Bei der Überprüfung der Voraussetzungen durch die Setup-Routine kam es zu einer Warnmeldung, da der IIS ohne ein SSL-Zertifikat betrieben wird. Diese Warnung wurde erwartet und nicht weiter berücksichtigt, da der Server gemeinsam mit den XenApp Servern im sicheren internen Netz betrieben wird und der unverschlüsselte Zugriff auf die EdgeSight Website darüber hinaus ohnehin nur lokal auf dem Server erfolgen soll. Das Setup wurde mit benutzerdefinierten Einstellungen durchgeführt. Als Name für das Unternehmen, d. h. den ersten Kunden im Kontext von EdgeSight, wurde „UMSICHT“ verwendet. Als Superuser für den EdgeSight Server wurde der Administrator definiert, wobei an dieser Stelle zu berücksichtigen ist, dass das hier vergebene Kennwort unabhängig vom Passwort des entsprechenden Domänen-Benutzers ist und nur innerhalb des EdgeSight-Systems Verwendung findet. Entsprechend sorgfältig ist es zu dokumentieren und aufzubewahren.

Im nächsten Dialogschritt wurde der Zugang zur Microsoft Exchange E-Mail Infrastruktur des Unternehmens konfiguriert, so dass der EdgeSight Server in der Lage ist, Statusmeldungen per E-Mail zu verschicken. Als Lizenz-Server kam die bereits vorhandene Citrix Licensing Installation der XenApp Farmen zum Einsatz, die auf Grund der eingesetzten XenApp Lizenzen in der Platinum Edition über passende EdgeSight-Lizenzen verfügt. Daraufhin konnte die grundlegende Konfiguration des EdgeSight Servers vorgenommen werden, in deren Verlauf im ersten Schritt der Zugriff auf die Reporting Services einzurichten war.

Weiterhin konnte innerhalb der Unternehmenskonfiguration eine neue Workerkonfiguration angelegt werden, die verschiedene Wartungsaufgaben innerhalb der EdgeSight Infrastruktur steuert. Diese neue Konfiguration erhielt den Namen „UMSICHT XenApp Standard“ und wurde weitestgehend mit Standardwerten parametrisiert:

- Die Anlagenliste für die zu verwaltenden Geräte wird erstellt, wenn i. d. R. keine Benutzer aktiv sind, täglich um 5:00 Uhr.
- Die Erfassung des freien Speicherplatzes auf den Laufwerken der Server erfolgt dagegen zu einem Zeitpunkt, an dem möglichst viele Benutzer an den Systemen angemeldet sind, täglich um 13:00 Uhr.
- Der Ausfallbericht, in dem Fehlerzustände des Systems erfasst werden, wird dagegen wiederum außerhalb der Kernarbeitszeit bereinigt, täglich um 23:45 Uhr.
- Die lokale Datenbank der Agenten wird ebenfalls außerhalb der Kernarbeitszeit gewartet, um im produktiven Betrieb möglichst keine zusätzliche Last zu erzeugen, täglich um 20:30 Uhr.
- Die Prüfung auf Änderung der Konfiguration durch die Agenten erfolgt im laufenden Betrieb kontinuierlich.
- Die Agenten laden ihre lokal zwischengespeicherten Daten einmal täglich auf den EdgeSight Server hoch und zwar um 19:30 Uhr.

Nach der Anzeige einer Zusammenfassung der Konfiguration erfolgte deren Zuweisung an alle XenApp-Server. Ebenfalls unter dem Namen „UMSICHT XenApp Standard“ wurde daraufhin eine Konfiguration für die Eigenschaften der XenApp-Agenten angelegt. Diese wurden so parametrisiert, dass sie kontinuierlich Daten erfassen. Es werden dabei die vorhandenen XenApp Lizenzen genutzt, um den verfügbaren Funktionsumfang festzulegen. Für die erweiterten Einstellungen kommen Standard-Werte zum Einsatz. Auch die Konfiguration der Agenten-Eigenschaften wurde daraufhin an alle XenApp-Server zugewiesen. Die Zuweisungen fanden sich daraufhin innerhalb der Geräteverwaltung bei den Abteilungen wieder.

Auch für die Konfiguration des Echtzeit-Dashboards, die über die Schaltfläche „Neue Echtzeitkonfiguration“ erzeugt wurde, kam der Name „UMSICHT XenApp Standard“ zum Einsatz. Die Zuweisung von Geräten wurde allerdings zunächst abgebrochen, da noch keine Agenten auf den Terminal Servern vorhanden und diese im Kontext von EdgeSight somit noch nicht bekannt waren.

12.4.3 XenApp 6.5/EdgeSight Agenten 5.4

Um die Agenten installieren zu können, wurden für die per Citrix Provisioning Services bereitgestellten Terminal Server jeweils neue Versionen des Standard-Images angelegt und im Maintenance-Modus gestartet, so dass Schreibzugriffe auf das Image möglich waren.

Unter XenApp 6.5 auf dem Windows Server 2008 R2 wurde der Citrix EdgeSight Agent in der Version 5.4.9.10 (x64) installiert. Auch in diesem Fall wurde als Unternehmen entsprechend der Konfiguration auf dem EdgeSight Server „UMSICHT“ angegeben. Eine Abfrage des Funktionsumfangs – Standard oder Advanced – erfolgte auch hier nicht, da der Agent automatisch in der Advanced-Version installiert wird, wenn XenApp Lizenzen der Platinum Edition vorhanden sind. Die EdgeSight-Daten werden auf die Cache-Disks der provisionierten Terminal Server in den Pfad „E:\Citrix\System Monitoring\Data“ geschrieben, so dass die gesammelten Informationen persistent erhalten bleiben und bei einem Neustart der provisionierten Server nicht verloren gehen.

Der EdgeSight Server wurde mit seiner IP-Adresse angegeben und die Windows-Firewall des Systems automatisch angepasst. Abschließend wurde im Computer-spezifischen Teil der Registry (HKEY_LOCAL_MACHINE) der Registry-Key „SOFTWARE\Wow6432Node\Citrix\System Monitoring\Agent\Core\4.00\RemoteSecurity“ auf den Wert „0“ geändert, um dem Administrator vom EdgeSight Server aus das Abfragen der lokalen EdgeSight-Daten zu ermöglichen. Damit war die Installation des Agenten abgeschlossen. Nach einem Neustart wurde der Terminal Server heruntergefahren und diese Version des Provisioning Server Images in den Produktivbetrieb übernommen.

12.4.4 Inbetriebnahme der Agents und Troubleshooting

Vor dem Neustart aller Terminal Server mit den neuen Image-Versionen wurden gemäß Vorgabe seitens Citrix diverse Pfade vom Scannen durch den auf den Terminal Servern und dem EdgeSight Server vorhandenen Antivirus-Client ausgenommen, um die Stabilität und

Performance der EdgeSight Infrastruktur nicht zu beeinträchtigen. Die entsprechenden Einstellungen wurden über die zentrale Server-Komponente des Antivirus-Clients verteilt. Dies betraf auf dem EdgeSight Server die folgenden Pfade:

- `C:\Program Files (x86)\Citrix\System Monitoring\Server\EdgeSight\scripts\rssh`
- `C:\Program Files (x86)\Citrix\System Monitoring\Server\EdgeSight\Pages`
- `C:\Program Files\Microsoft SQL Server\MSRS10_50.MSSQLSERVER\Reporting Services`
- `C:\Program Files\Microsoft SQL Server\MSSQL10_50.MSSQLSERVER\MSSQL\DATA`
- `C:\Windows\System32\LogFiles`

Auf den Windows Server 2008 R2 Systemen mit XenApp 6.5 betrafen die Ausschlüsse die folgenden Prozesse und Pfade:

- `C:\Program Files (x86)\Citrix\System Monitoring\Agent\Core\rscorsvc.exe`
- `C:\Program Files (x86)\Citrix\System Monitoring\Agent\Core\Firebird\bin\fbserver.exe`
- `E:\Citrix\System Monitoring\Data`

Daraufhin wurden alle Terminal Server neu gestartet, so dass sie auf die aktuelle Image-Version mit dem EdgeSight Agenten wechselten. Dieser nahm nun auf allen Systemen die Arbeit auf und meldete die Server automatisch beim EdgeSight Server, was dieser per E-Mail quittierte. Im Nachgang traten allerdings Probleme auf, die sich darin äußerten, dass der EdgeSight Agent sporadisch den Dienst versagte. Im Fehlerfall meldete der EdgeSight Server eine Betriebswarnung für den jeweiligen Agenten (vgl. Abbildung 12-6).

Abbildung 12-6: EdgeSight Betriebswarnung

Quelle: Eigene Darstellung

Daraufhin ließ sich der betroffene Agent nicht mehr abfragen und übermittelte auch keine Daten mehr an den EdgeSight Server. Eine Untersuchung des Problems ergab, dass der Fehler nicht von den Agenten, sondern vom EdgeSight Server selbst verursacht wurde. Dieser loggte im Ereignisprotokoll „Anwendung" zahlreiche Warnungen mit der Ereignis-ID 1309. Als Ursache konnte ein Fehler[121] im Zusammenhang mit EdgeSight identifiziert werden. Das

[121] Event ID 1309 after Installing MS11-100 on the EdgeSight Server: http://support.citrix.com/article/CTX132116 (abgerufen am 01.02.2016).

Problem ließ sich durch einen Eintrag in der Datei „C:\Program Files (x86)\Citrix\System Monitoring\Server\EdgeSight\Pages\web.config“ auf dem EdgeSight Server beheben:

```
(…)

      <!-- Ops pages derive from AdminPage -->
      <location path="app/ops">
            <system.web>
            <pages pageBaseType="Citrix.EdgeSight.Web.AdminPage, Citrix.EdgeSight.Web"…
            </system.web>
      </location>

      <appSettings>
            <add key="ReportViewerServerConnection" value="Citrix.EdgeSight.W…
            <add key="ReportViewerMessages" value="Citrix.EdgeSight.Web.Repor…
            <add key="aspnet:MaxHttpCollectionKeys" value="10000"/>
      </appSettings>
</configuration>
```

Nach einem Neustart des IIS auf dem EdgeSight Server sowie aller betroffenen Terminal Server trat das Problem nicht mehr auf und die EdgeSight Infrastruktur lief stabil.

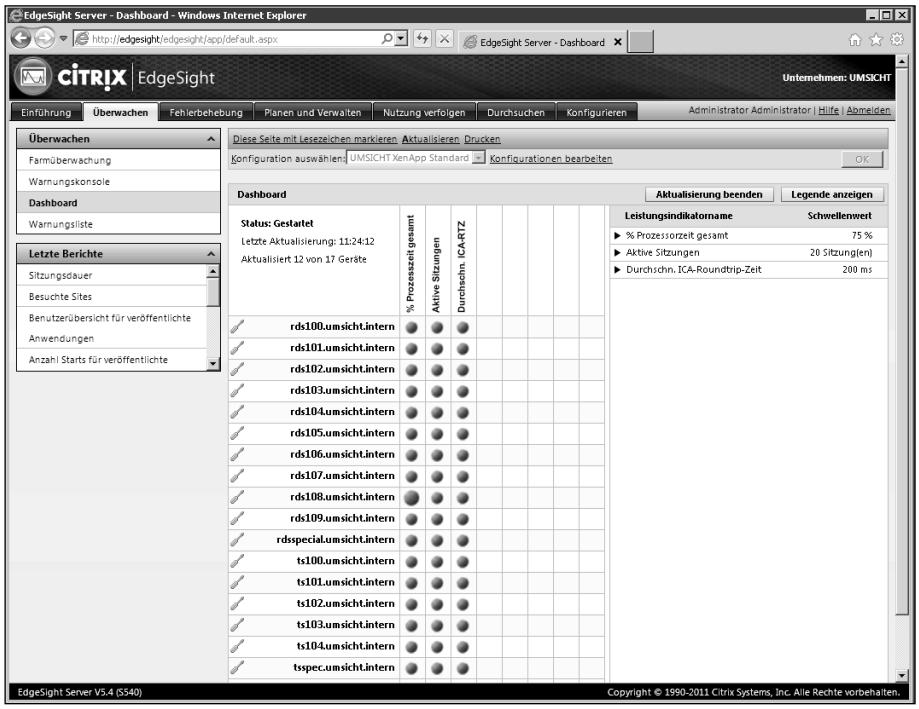
12.5 Monitoring, Berichte und Auswertung der Daten

Nach der Basisinstallation und -konfiguration begann die EdgeSight Infrastruktur mit der Erfassung von Daten, wobei dies nach dem Prinzip erfolgt, zunächst sämtliche verfügbaren Daten zu erfassen und die Auswertung sowie Eingrenzung der gewünschten Datensätze im Nachgang über die EdgeSight Konsole vorzunehmen.

12.5.1 Echtzeit-Monitoring

Nachdem die Clients sich beim Server angemeldet hatten, konnte zunächst die Konfiguration des „Echtzeit-Dashboards“ abgeschlossen werden. Aus einer Übersicht, die alle bekannten Clients mit ihrer jeweiligen EdgeSight-Version ausweist, wurden alle Terminal Server als Konfigurationsmitglieder des Dashboards aufgenommen. Im nächsten Schritt waren die gewünschten Leistungsindikatoren zu wählen, die im Dashboard angezeigt werden sollen. Für die Basiskonfiguration wurden hier die gesamte Prozessorzeit mit einer Warnschwelle von 75 Prozent, die Anzahl aktiver Sitzungen pro Server mit einem Schwellwert von 20 sowie die Roundtrip-Zeit von ICA-Paketen im Netzwerk mit einer Warnschwelle von 200 ms gewählt. Anschließend konnte das Ergebnis auf der Seite „Überwachen\Dashboard“ angezeigt werden. Für alle zu überwachenden Systeme werden die Leistungsindikatoren nach Art einer Ampel über Punkte in den Farben Grün, Gelb und Rot angezeigt, so dass aktuelle Probleme auf einen Blick zu erkennen sind (vgl. Abbildung 12-7).

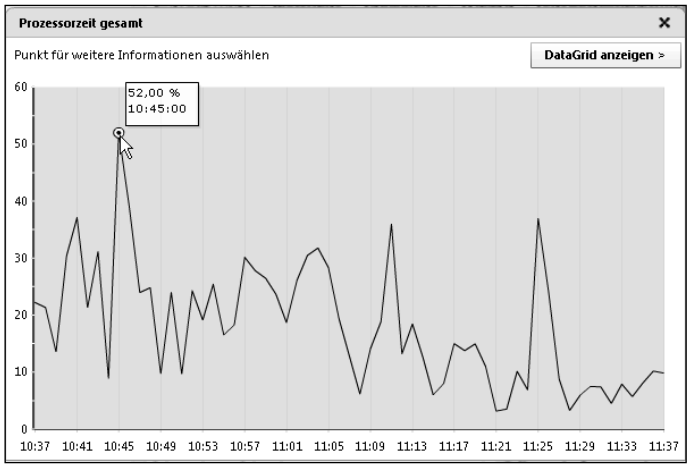
Abbildung 12-7: Echzeit-Monitoring im EdgeSight-Dashboard



Quelle: Eigene Darstellung

Über das „Werkzeug"-Symbol kann daraufhin die Detailansicht einzelner Server aufgerufen werden. Ein Doppelklick auf einen der Leistungsindikatoren aus der Systemübersicht führt zu einem detaillierteren Graphen des gewünschten Parameters, beispielsweise der Prozessorzeit (vgl. Abbildung 12-8).

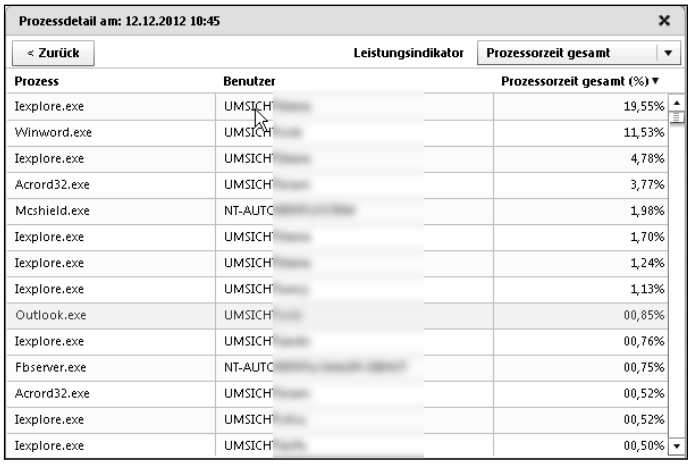
Abbildung 12-8: Echtzeit-Monitoring der Prozessorlast



Quelle: Eigene Darstellung

Mit einem Klick auf einzelne Punkte des Graphen kann ermittelt werden, welche Prozesse zu der jeweiligen Auslastung beigetragen haben (vgl. Abbildung 12-9).

Abbildung 12-9: Detail-Auswertung der Prozessorlast



Quelle: Eigene Darstellung

Ebenso ist es möglich, über die Prozessliste im Hauptfenster in Erfahrung zu bringen, welche Prozesse aktuell ausgeführt werden und wie deren Leistungsindikatoren im Detail beschaffen sind. Über die Sitzungsliste ist eine solche Auswertung weiterhin pro Benutzer möglich. Über das Dashboard und die Berichte im Bereich „Fehlerbehebung" lassen sich somit kurzfristige Auswertungen des aktuellen Status und der lokal auf den Clients zwischengespeicherten Daten realisieren.

12.5.2 Mitgelieferte Berichte

Die Auswertung der mittel- bis langfristigen Daten, welche zentral in der EdgeSight Datenbank gespeichert sind, erfolgt über die Reporting Services des SQL Servers. Hierzu sind zahlreiche vorgefertigte Berichte im Umfang von EdgeSight enthalten, die weit über die Möglichkeiten des früheren Resource Managers hinausgehen. So kann im Bereich „Nutzung verfolgen" beispielsweise mit dem Bericht „Citrix Lizenzierung\Lizenznutzungstrends" die Nutzung der vorhandenen Lizenzen über eine längere Periode ermittelt werden (vgl. Abbildung 12-10).

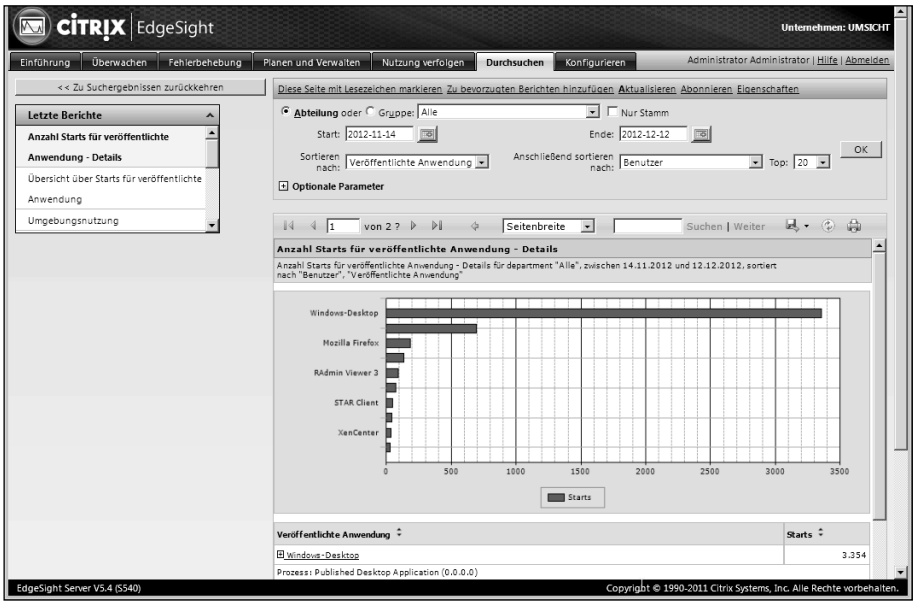
Die Übersicht über sämtliche EdgeSight-Berichte[122] findet sich im Bereich „Durchsuchen", wo über die Optionen in der linken Fensterhälfte die Suche nach bestimmten Berichten eingegrenzt werden kann. Auf diesem Weg können detaillierte Informationen zur Nutzung von veröffentlichten Anwendungen erhoben werden. So gibt der Bericht „Anzahl Starts für veröffentlichte Anwendung – Details" eine grafisch aufbereitete Übersicht der meist genutzten Applikationen für einen Zeitraum aus.

122 EdgeSight 5.4 Report List: http://community.citrix.com/display/edgesight/EdgeSight+5.4+Report+List (abgerufen am 01.02.2016).

Abbildung 12-10: Auswertung der Lizenznutzung

Quelle: Eigene Darstellung

Abbildung 12-11: Auswertung der Nutzung veröffentlichter Anwendungen



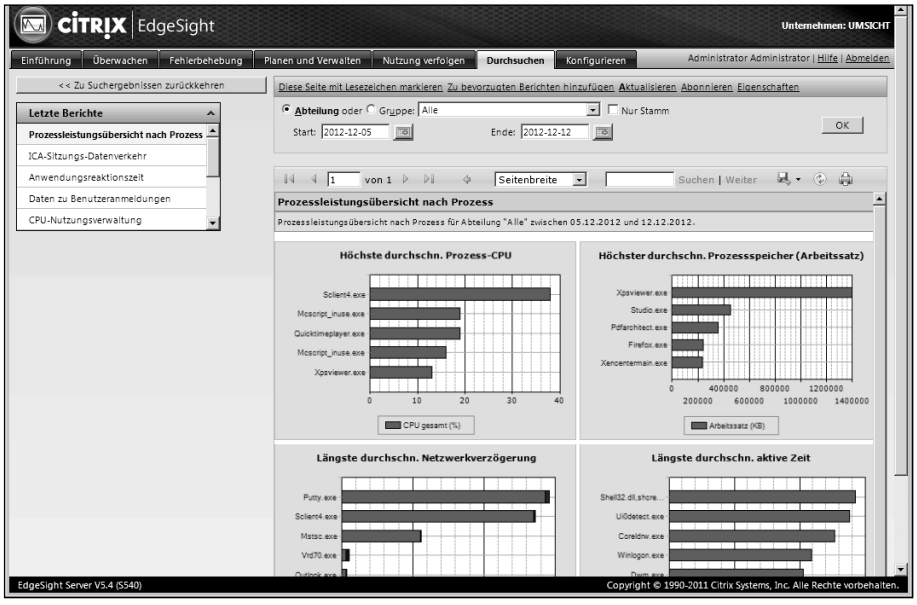
Quelle: Eigene Darstellung

Der Bericht „Übersicht über Starts für veröffentlichte Anwendung“ bricht eben diese Informationen auf eine tagesgenaue tabellarische Ansicht herunter. Diese Ansicht ist allerdings bei längeren Betrachtungszeiträumen von mehreren Wochen in der EdgeSight-Console kaum mehr lesbar. Daher bietet sich hier zur weiteren Auswertung der Export nach Microsoft Excel an.

Im Hinblick auf das Kapazitätsmanagement sind insbesondere die Berichte aus der Kategorie „Leistung“ interessant. Diese können helfen, typische Probleme und Fehler zu identifizieren, welche die vom Benutzer wahrgenommene Arbeitsgeschwindigkeit in Terminal Server Sitzungen beeinträchtigen können. Exemplarisch seien genannt:

- Client- und Serverstartdauer für Sitzung: Der Bericht weist getrennt nach Client und Server die Zeitspanne aus, die beim Aufbau einer Sitzung vergeht. Dies kann helfen Probleme bei der Anmeldung von Benutzern zu diagnostizieren.
- CPU-Nutzungsverwaltung: Der Bericht zeigt pro Server, ob das Citrix-eigene CPU-Management aktiv ist oder nicht und welche Prozesse durch das CPU-Management gedrosselt wurden.
- Prozessleistungsübersicht nach Prozess: Der Bericht identifiziert in grafisch aufbereiteter Form die CPU- und RAM-intensivsten Prozesse über einen definierbaren Zeitraum (vgl. Abbildung 12-12).

Abbildung 12-12: Prozessleistungsübersicht



Quelle: Eigene Darstellung

Weitere Berichte betreffen den Netzwerkbereich, dort besonders das ICA-Protokoll und somit die vom Benutzer wahrgenommene Arbeitsgeschwindigkeit bis hin zum Client.

12.5.3 EdgeSight-Datenbankschema

Sollte eine Fragestellung mit Hilfe der integrierten Berichte nicht zu beantworten sein, besteht die Möglichkeit, angepasste Berichte – sogenannte „Custom Reports" – zu erstellen. Hierzu finden sich im Citrix Developer Network (CDN) einige Custom Reports (vgl. [Citrix, 2009], S. 254ff.) sowie in der Citrix Knowlegde Base eine Einführung in die Struktur angepasster Berichte (vgl. [Citrix, 2007]), letztere allerdings leider nur mit Bezug auf die ältere EdgeSight-Version 4.5.

12.5.3.1 Tabellen

Einen guten Einstiegspunkt in das Datenbankschema von EdgeSight bis hin zum Aufbau eigener Reports bietet eine mehrteilige Serie von Blog-Beiträgen[123]. Reports sind insbesondere dann von Interesse, wenn die komplette Abfrage-Logik inklusive Aufbereitung und Darstellung der Ergebnisse direkt in die EdgeSight Konsole integriert werden soll. Die Auswertung der in der EdgeSight Datenbank erfassten Informationen kann jedoch auch mittels Ad-hoc Abfragen direkt gegen die Datenbank erfolgen, was mit dem Microsoft SQL Server Management Studio oder geeigneten Tools von Drittanbietern möglich ist.

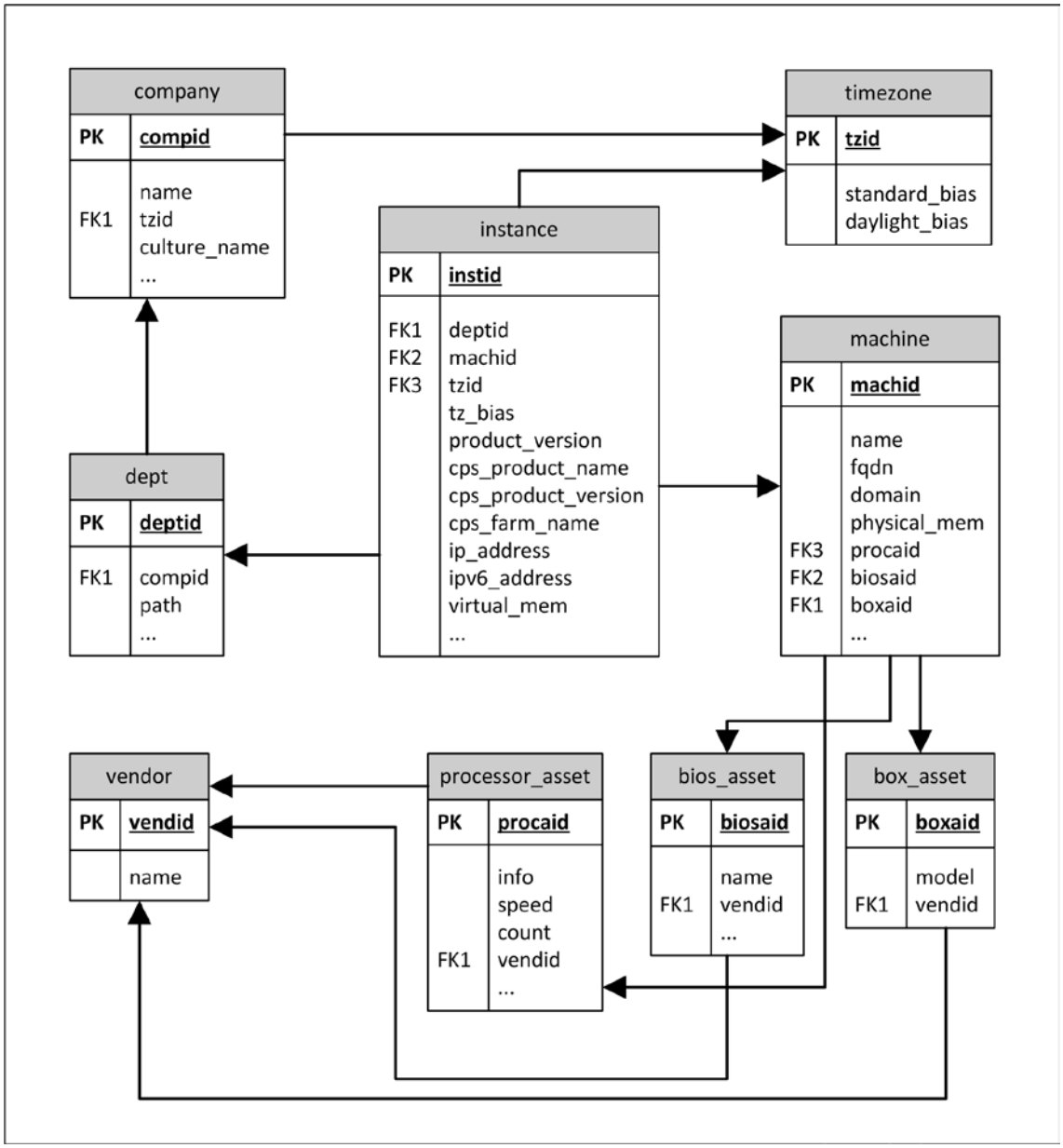
Hierfür ist ein grundlegendes Verständnis des zu Grunde liegenden Datenbankschemas nützlich. Das Schema ist allerdings seitens des Herstellers weitestgehend undokumentiert. Entsprechend werden im Folgenden die im Kontext dieser Arbeit relevanten Teile des Schemas sowie Beispielabfragen erläutert.

Es existieren diverse Tabellen innerhalb der Datenbank, welche die Organisationsstruktur der überwachten Farm(en) sowie Basisdaten der darin enthaltenen Maschinen bereitstellen (vgl. Abbildung 12-13). In der Tabelle „company" können mehrere Mandanten gepflegt werden. Dabei kann sich um unterschiedliche Unternehmen oder aber – beispielsweise innerhalb eines Konzerns – um Unternehmensteile handeln. Zu einem Mandanten werden u. a. Name, Sprachraum („culture_name", z. B. de-DE) sowie Zeitzone[124] erfasst. Letztere wird allerdings pro Maschine noch einmal separat verwaltet. Abteilungen sind in der Tabelle „dept" gespeichert. Dabei handelt es sich im Kontext von EdgeSight jedoch nicht um eine Abbildung der Organisationsstruktur des Unternehmens, sondern um die Struktur der in EdgeSight erfassten Citrix-Farm(en).

[123] Building Custom EdgeSight Reports:
http://blogs.sepago.de/d/nicholas/2010/07/16/building-custom-edgesight-reports-part-2-the-database
http://blogs.sepago.de/d/nicholas/2010/07/26/building-custom-edgesight-reports-part-3-the-query
http://blogs.sepago.de/d/nicholas/2010/09/27/building-custom-edgesight-reports-part-4-the-wedding (drei Links, abgerufen am 01.02.2016).

[124] Zeitstempel werden in der EdgeSight Datenbank grundsätzlich als Koordinierte Weltzeit (Coordinated Universal Time (UTC)) gespeichert. Mit Hilfe der Angaben zur Zeitzone können in Berichten korrigierte Zeitstempel passend zur jeweiligen Zeitzone des EdgeSight Agenten ausgegeben werden, was insbesondere bei über mehrere Zeitzonen verteilten Agenten notwendig ist. Da sich alle Agenten des Beispielszenarios in einer Zeitzone befinden, bleibt diese Option in den weiteren Betrachtungen unberücksichtigt und die Zeitstempel werden innerhalb der Ad-hoc Abfragen manuell korrigiert.

Abbildung 12-13: EdgeSight ER Diagramm – Organisation



Quelle: Eigene Darstellung

Eine Abteilung ist somit eine logische Gruppierung von Agenten innerhalb eines Mandanten, die mit einem definierten Satz an Konfigurationen, Regeln und Alarmen versehen wird. Die Wurzel-Abteilung „All“ sowie Unterabteilungen pro XenApp-/XenDesktop-Farm werden bei der Installation standardmäßig angelegt und können nicht gelöscht oder verändert werden (vgl. [Hasan, 2013], S 19ff.).

Tabelle 12-1: Ad-hoc Abfrage der Agenten-Maschinen

```
SELECT
     instance.instid AS "ID",
     machine.name AS "Client",
     dept.name AS "Department",
     machine.mem_physical AS "Phys. RAM",
     instance.virtual_mem AS "Virt. RAM",
     processor_asset.info AS "Processor",
     processor_asset.count AS "Proc.#",
     bios_asset.name AS "BIOS",
     box_asset.model AS "Box",
     tz_bias AS "UTC+"
FROM
     instance JOIN
     machine ON instance.machid = machine.machid JOIN
     dept ON dept.deptid = instance.deptid JOIN
     processor_asset ON processor_asset.procaid = machine.procaid JOIN
     bios_asset ON bios_asset.biosaid = machine.biosaid JOIN
     box_asset ON box_asset.boxaid = machine.biosaid
```

Quelle: Eigene Darstellung

Die Daten zu Servern und Desktops sind auf zwei miteinander verknüpfte Tabellen verteilt. Die Tabelle „instance“ beinhaltet dabei Informationen zu Konfiguration und Citrix-Software, darunter die IP-Adressen des jeweiligen Systems, die Version des EdgeSight Agenten („product_version“), Daten zur installierten Version von Citrix XenApp oder XenDesktop („cps_product_name“, „cps_product_version“ und „cps_farm_name“), sowie den virtuellen Hauptspeicher („virtual_mem“). Informationen, wie der physische Hauptspeicher („physical_mem“), Name, Full Qualified Domain Name („fqdn“) oder der NetBIOS-Name der Active Directory Domain („domain“) finden sich in der Tabelle „machine“.

Weitere Hardware-nahe Informationen werden mittels der Tabellen „processor_asset“, „bios_asset“ und „box_asset“ referenziert, die ihrerseits wiederum auf die Tabelle „vendor“ mit dem Namen des jeweiligen Herstellers verweisen, soweit dieser erkennbar ist. Mit Hilfe dieser Tabellen können Erkenntnisse zur Ausstattung und Ausprägung (physisch oder virtuell) der Agenten-Maschinen gewonnen werden. So liefert eine Ad-Hoc Abfrage (vgl. Tabelle 12-1) gegen die EdgeSight Datenbank eine tabellarische Aufstellung aller EdgeSight Clients mitsamt ihrer Ausstattung an Hauptspeicher und Prozessoren sowie dem Offset gegenüber der Koordinierten Weltzeit. Anhand der Beispielausgabe (vgl. Tabelle 12-2) ist erkennbar, dass es sich bei den Clients offensichtlich um auf XenServer Hosts laufende virtuelle Maschinen handelt.

Tabelle 12-2: Ausgabe der Ad-hoc Abfrage zu Agenten-Maschinen

ID	Client	Department	Phys. RAM	Virt. RAM	Processor	Proc.#	BIOS	Box	UTC+
63	RDS100	XenApp65…	12578136	2097024	Intel(R) Xeon(R)...	4	Xen - 0	HVM domU	120
64	RDS101	XenApp65…	12578136	2097024	Intel(R) Xeon(R)...	4	Xen - 0	HVM domU	120
65	RDS102	XenApp65…	12578136	2097024	Intel(R) Xeon(R)...	4	Xen - 0	HVM domU	120

ID	Client	Department	Phys. RAM	Virt. RAM	Processor	Proc.#	BIOS	Box	UTC+
66	RDS103	XenApp65…	12578136	2097024	Intel(R) Xeon(R)...	4	Xen - 0	HVM domU	120
…	…	…	…	…	…	…	…	…	…

Quelle: Eigene Darstellung

Da das Datenbankschema sehr komplex und öffentlich nicht dokumentiert ist, sollten über die Bestandsinformationen hinaus Abfragen über die Performancedaten von Servern und Sitzungen nicht direkt gegen die Tabellen der Datenbank, sondern gegen zu diesem Zweck angepasste Ansichten auf die Tabellen erfolgen.

12.5.3.2 Ansichten (Views)

Die 96 Ansichten von EdgeSight bereiten die Informationen der Datenbank für Ad-hoc Abfragen mit verschiedenen Schwerpunkten auf – je nachdem, ob ein Server als Ganzes, einzelne Prozesse, Anwendungen, Benutzer oder Clients im Fokus stehen.

So enthält die Ansicht „vw_system_perf" allgemeine Daten zur Performance der Server, darunter Informationen über Prozessor, Hauptspeicher und Festplatten. Die Ansicht „vw_ctrx_system_perf" beinhaltet zusätzliche Informationen im Hinblick auf den Einsatz als Terminal Server, wie die Anzahl aktiver Sessions. Und die Ansichten „vw_image_perf" und „vw_ctrx_session_perf" brechen die Performancedaten auf die Ebene einzelner Prozesse und Sitzungen herunter.

Die einzelnen Datensätze in den Ansichten beinhalten mit dem Feld „dtperiod" jeweils einen Zeitstempel, der den Beginn eines Messintervalls im Stundentakt markiert. Für die einzelnen Messobjekte sind im Datensatz jeweils die Summe (*_sum) sowie die Anzahl (*_cnt) und für manche Objekte auch das Maximum (*_peak) der im Intervall erfassten Messwerte gespeichert. Über das Attribut „instid" der Tabelle „instance" wird der Bezug zwischen den Ansichten sowie den EdgeSight Agenten und Terminal Servern hergestellt (vgl. Abbildung 12-14).

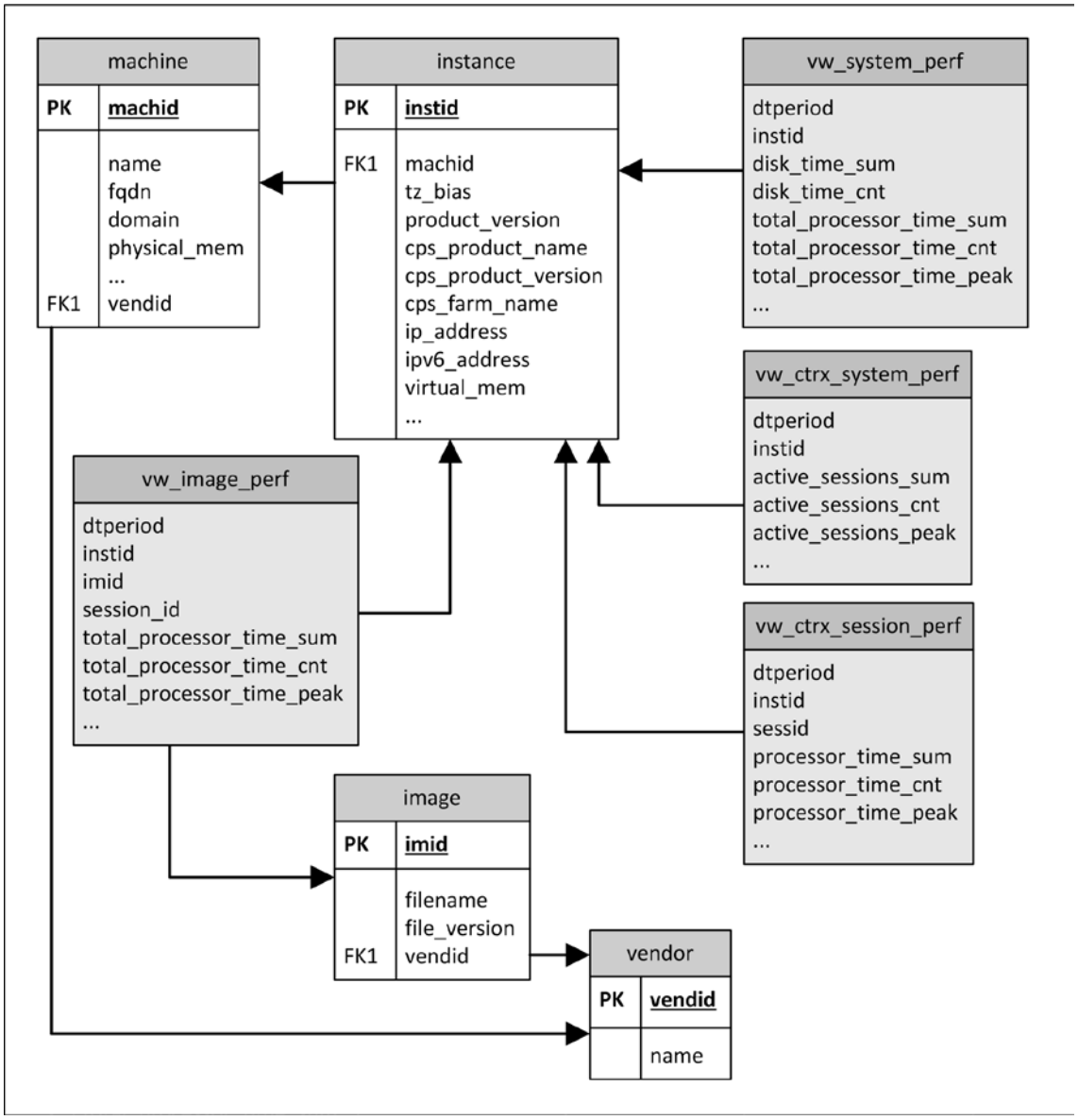
Im Fall der Ansicht „vw_image_perf" sind zusätzlich Informationen aus den Tabellen „image" und „vendor" nötig, um zu verwertbaren Aussagen über einzelne Prozesse zu gelangen.

Für alle Ansichten mit direktem Bezug zum Betrieb als Citrix XenApp Terminal Server (vw_ctrx_*) eröffnen die korrespondierenden Archiv-Ansichten (vw_ctrx_archive_*) einen einfacheren Weg, um zu für Menschen lesbaren Auswertungen der Performancedaten zu gelangen.

Zweck der Archiv-Ansichten ist es, „Daten, die langfristig genutzt werden sollen, z. B. für Trendanalysen, zur Belegung der Lizenzeinhaltung und zur Kapazitätsplanung" (vgl. [Citrix, 2011a], S. 77ff.) im Rahmen von Data Warehousing zu archivieren und in sekundären Datenbanken oder anderweitigen Speichersystemen zu sichern, um die primäre Datenbank zu entlasten.

Auf diese Weise stehen die gewünschten Daten dauerhaft zur Verfügung, ohne dass die Optimierungseinstellungen der EdgeSight-Datenbank dahingehend gelockert werden müssten, dass die Daten länger als 30 oder gar 90 Tage in der primären Datenbank vorgehalten würden.

Abbildung 12-14: EdgeSight-Datenbankansichten



Quelle: Eigene Darstellung

Die Archiv-Ansicht erweitert dazu die zu Grunde liegende Ansicht um das Feld „inserted_date“, das den Zeitpunkt beinhaltet, an dem der jeweilige Datensatz in die Datenbank geschrieben wurde. Somit können Data Warehousing Prozesse die Daten identifizieren, die nach einem bestimmten Zeitpunkt, d. h. nach dem letzten Data Warehousing Vorgang, geschrieben wurden, und gezielt nur diese Datensätze archivieren. In einer Archiv-Ansicht wird weiterhin das Feld „dtperiod“ zu „time_stamp“ umbenannt.

Zusätzlich stellt die Archiv-Ansicht den Bezug zu weiteren Tabellen her und reichert die reinen Performancedaten um Bestandsdaten an. Während die Basis-Ansicht allein i. d. R. nicht für aussagekräftige Auswertungen genügt, gelingen diese mit der darauf aufbauenden Archiv-Ansicht, womit letztere nicht nur für Data Warehousing Konzepte, sondern auch für Ad-hoc Abfragen geeignet ist.

Das Prinzip sei im Folgenden anhand der Ansicht „vw_ctrx_system_perf" sowie der zugehörigen Archiv-Ansicht „vw_ctrx_archive_system_perf" erläutert. Erstere bezieht sich ausschließlich auf die Tabelle „ctrx_system_perf" und übersetzt die Spalten dieser Tabelle auf aussagekräftige Bezeichner (vgl. Tabelle 12-3). Die Ansicht gibt dabei jedoch ausschließlich die Performancedaten zu einem konkreten Zeitpunkt (dtperiod) für einen bestimmten EdgeSight Agenten (instid) aus, ohne dass aus dem Ergebnis im Rahmen einer Ad-hoc Abfrage ersichtlich wäre, um welchen Server es sich jeweils handelt.

Tabelle 12-3: Abfrage auf „vw_ctrx_system_perf"

```
SELECT
      dtperiod,
      instid,
      i1 AS active_sessions_sum,
      i2 AS active_sessions_cnt,
      i3 AS active_sessions_peak,
      i4 AS inactive_sessions_sum,
      i5 AS inactive_sessions_cnt,
      i6 AS inactive_sessions_peak,
      i7 AS total_sessions_sum,
      i8 AS total_sessions_cnt,
      i9 AS total_sessions_peak
FROM
      dbo.ctrx_system_perf
```

Quelle: Eigene Darstellung, basierend auf dem SQL-Code der EdgeSight Datenbank von Citrix Systems

Die korrespondierende Archiv-Ansicht „vw_ctrx_archive_system_perf" erübrigt es, im Rahmen von Ad-hoc Abfragen selbst den Bezug zu Tabellen mit Bestandsdaten herzustellen, da die Informationen aus den Tabellen „company", „dept", „instance" sowie „machine" bereits enthalten sind (vgl. Tabelle 12-4).

Für die übrigen Ansichten mit dem Namensschema „vw_ctrx_*" verhält es sich entsprechend, so dass generell die Archiv-Ansichten der geeignetere Weg sind, um eigene Auswertungen von in EdgeSight erfassten Daten zu erstellen. Im Folgenden werden die entsprechenden Ad-hoc Abfragen zu den in Kapitel 6.5 relevanten Fragestellungen erläutert.

Tabelle 12-4: Abfrage auf „vw_ctrx_archive_system_perf"

```
SELECT
      sp.dtperiod AS time_stamp,
      sp.inserted_date,
      c.compid,
      d.deptid,
      m.machid,
      sp.instid,
      c.name AS company_name,
      d.name AS dept_name,
      d.path AS dept_path,
      m.name AS machine_name,
      m.domain AS machine_domain,
      i.cps_product_name AS xen_product_name,
      i.cps_product_version AS xen_product_version,
      i.cps_product_service_pack AS xen_product_service_pack,
      i.cps_farm_name AS xen_farm_name,
      sp.i1 AS active_sessions_sum,
      sp.i2 AS active_sessions_cnt,
      sp.i3 AS active_sessions_peak,
      sp.i4 AS inactive_sessions_sum,
      sp.i5 AS inactive_sessions_cnt,
      sp.i6 AS inactive_sessions_peak,
      sp.i7 AS total_sessions_sum,
      sp.i8 AS total_sessions_cnt,
      sp.i9 AS total_sessions_peak
FROM
      dbo.ctrx_system_perf AS sp INNER JOIN
      dbo.instance AS i ON i.instid = sp.instid INNER JOIN
      dbo.machine AS m ON m.machid = i.machid INNER JOIN
      dbo.company AS c ON c.compid = i.compid INNER JOIN
      dbo.dept AS d ON d.deptid = i.deptid
```

Bestandsdaten
Performancedaten

Quelle: Eigene Darstellung, basierend auf dem SQL-Code der EdgeSight Datenbank von Citrix Systems

12.5.4 Ad-hoc Abfragen

Für einen Einstieg in die Analyse der in der EdgeSight Datenbank enthaltenen Informationen seien die folgenden Abfragen, die alle ad-hoc aus dem Microsoft SQL Server Management Studio heraus ausgeführt werden können, empfohlen:

Die Abfrage (Tabelle 12-5) liefert eine Übersicht aller in der Datenbank existierenden Views.

Tabelle 12-5: Abfrage aller Ansichten

```
SELECT name
FROM sys.views
```

Quelle: Eigene Darstellung

Die nachstehende Abfrage (Tabelle 12-6) gibt alle Spalten einer bestimmten Ansicht aus.

Tabelle 12-6: Abfrage der Spalten einer Ansicht

```
1  SELECT column_id,name
2  FROM sys.columns
3  WHERE object_id = OBJECT_ID('dbo.<Name-der-gewünschten-Ansicht>')
```

Quelle: Eigene Darstellung

Für den Einstieg in die statistische Auswertung des Nutzerverhaltens liefert die folgende Abfrage (Tabelle 12-7) die Gesamtzahl aller über den kompletten Beobachtungszeitraum erfassten Sitzungen.

Tabelle 12-7: Abfrage der Gesamtzahl aller Sessions

```
1  SELECT COUNT(DISTINCT(sessid))
2  FROM vw_ctrx_archive_session_perf
```

Quelle: Eigene Darstellung

Diese können weiterhin nach ihrer Verteilung über die Zeit ausgewertet werden (Tabelle 12-8).

Tabelle 12-8: Abfrage der zeitlichen Verteilung der Sessions

```
1  SELECT
2       CONVERT(varchar(19), DATEADD(mi, timezone.daylight_bias,
              vw_ctrx_archive_system_perf.time_stamp), 120) AS "Timestamp",
3       SUM(vw_ctrx_archive_system_perf.active_sessions_sum /
              vw_ctrx_archive_system_perf.active_sessions_cnt) AS
              "TotalSessions"
4  FROM vw_ctrx_archive_system_perf INNER JOIN
5       instance ON vw_ctrx_archive_system_perf.instid = instance.instid
              INNER JOIN
6       timezone ON instance.tzid = timezone.tzid
7  GROUP BY vw_ctrx_archive_system_perf.time_stamp, timezone.daylight_bias
8  ORDER BY "Timestamp"
```

Quelle: Eigene Darstellung

Mittels des nachstehenden WHERE-Statements, welches vor dem GROUP BY-Statement eingefügt wird, kann der Beobachtungszeitraum eingegrenzt und beispielsweise nur auf Werktage sowie an jedem Tag auf die Zeit zwischen 10.00 und 14.00 Uhr beschränkt werden (Tabelle 12-9).

Tabelle 12-9: Begrenzung des Beobachtungszeitraums

```
    (…)
 7  WHERE
 8        DATEPART(dw, DATEADD(mi, timezone.daylight_bias,
               vw_ctrx_archive_system_perf.time_stamp)) <= 5 AND
 9        DATEPART(hh, DATEADD(mi, timezone.daylight_bias,
               vw_ctrx_archive_system_perf.time_stamp)) >= 10 AND
10        DATEPART(hh, DATEADD(mi, timezone.daylight_bias,
               vw_ctrx_archive_system_perf.time_stamp)) <= 14 AND
11        CONVERT(varchar(10), DATEADD(mi, timezone.daylight_bias,
               vw_ctrx_archive_system_perf.time_stamp), 111) >= '2014/06/23' AND
12        CONVERT(varchar(10), DATEADD(mi, timezone.daylight_bias,
               vw_ctrx_archive_system_perf.time_stamp), 111) <= '2014/07/20'
13  GROUP BY vw_ctrx_archive_system_perf.time_stamp, timezone.daylight_bias
14  ORDER BY "Timestamp"
```

Quelle: Eigene Darstellung

12.5.4.1 Veröffentlichte Anwendungen

Zwischen Sitzungen und veröffentlichten Applikationen besteht keine eineindeutige Relation. Dies liegt daran, dass ein Server mehrere veröffentlichte Applikationen hosten kann. Wenn nun ein Endanwender eine veröffentlichte Applikation startet, entsteht eine Sitzung auf einem entsprechenden Terminal Server. Wenn der Anwender dann – ohne die erste Anwendung zu schließen – eine weitere veröffentlichte Anwendung startet, die der gleiche Server bedienen kann, wird keine neue Sitzung erstellt, sondern stattdessen innerhalb der bereits bestehenden Sitzung die zweite Anwendung zusätzlich gestartet.

Tabelle 12-10: Abfrage nach veröffentlichten Applikationen

```
 1  SELECT
 2        published_application AS "Published Application",
 3        COUNT(DISTINCT(sessid)) AS "Usage"
 4  FROM
 5        vw_ctrx_archive_published_app_event INNER JOIN
 6        instance ON vw_ctrx_archive_published_app_event.instid = instance.instid
 7  INNER JOIN
 8        timezone ON instance.tzid = timezone.tzid
 9  WHERE
10        DATEPART(…)
11        DATEPART(…)
12        DATEPART(…)
13        CONVERT(…)
14        CONVERT(…)
15  GROUP BY
16        published_application
17  ORDER BY
18        Usage DESC
```

Quelle: Eigene Darstellung

Dies dient dem Verkürzen der Startzeiten sowie der effizienten Nutzung der zur Verfügung stehenden Ressourcen. Eine Instanz einer veröffentlichten Anwendung kann somit eindeutig einer Sitzung zugeordnet werden, eine Sitzung kann umgekehrt jedoch in Relation zu mehreren veröffentlichten Anwendungen stehen. Eine Abfrage (Tabelle 12-10) liefert die Anzahl der Instanzen veröffentlichter Applikationen und kann analog zur vorherigen Abfrage mit einem WHERE-Statement zeitlich begrenzt werden.

12.5.4.2 Client-Subnetze

Die folgende Abfrage (Tabelle 12-11) ermittelt, sortiert nach den Zeitstempeln der Einträge, die Anzahl der Sitzungen pro Client-Subnetz und gibt somit Auskunft darüber, von wo die Endanwender auf die Terminal Server zugreifen. Die Abfrage parst die ersten drei Bytes der IP-Adresse des Clients basierend auf der Annahme, dass die Clients in Class-C-Subnetzen organisiert sind. Bei Class-B- oder gar Class-A-Netzen ist die Abfrage entsprechend zu modifizieren.

Tabelle 12-11: Abfrage nach Sitzungen pro Client-Subnetz

```
SELECT
      CONVERT(varchar(19), DATEADD(mi, timezone.daylight_bias,
           vw_ctrx_archive_session_perf.time_stamp), 120) AS "Timestamp",
      CAST(PARSENAME(client_address, 4) AS NVARCHAR) + '.' +
      CAST(PARSENAME(client_address, 3) AS NVARCHAR) + '.' +
      CAST(PARSENAME(client_address, 2) AS NVARCHAR) + '.*' AS Network,
      COUNT(DISTINCT(sessid)) AS Sessions
FROM
      vw_ctrx_archive_session_perf INNER JOIN
      instance ON vw_ctrx_archive_session_perf.instid =
           instance.instid INNER JOIN
      timezone ON instance.tzid = timezone.tzid
GROUP BY
      CONVERT(varchar(19), DATEADD(mi, timezone.daylight_bias,
           vw_ctrx_archive_session_perf.time_stamp), 120),
      PARSENAME(client_address, 4),
      PARSENAME(client_address, 3),
      PARSENAME(client_address, 2)
ORDER BY
     "Timestamp"
```

Quelle: Eigene Darstellung

Auch diese Abfrage kann mittels des zuvor genannten WHERE-Statements zwischen dem FROM- und dem GROUP BY-Block zeitlich eingegrenzt werden. Das Ergebnis der Abfrage ist dann in eine Pivot-Tabelle zu überführen, wobei der Zeitstempel als Zeilenbeschriftung, das Netzwerk als Spaltenbeschriftung und die Sitzungen als Werte verwendet werden. Zusammengehörige Netzsegmente können dann weiterhin nach geografischen Standorten gruppiert und im Hinblick auf die absolute Anzahl an Sitzungen und deren relative Verteilung auf die Standorte ausgewertet werden.

12.5.4.3 Erweiterungen der EdgeSight-Datenbank

Für eine weitergehende Analyse der Last ist neben der geografischen Herkunft zu untersuchen, ob sich bezüglich der Bedienwünsche verschiedene Klassen von Anwendern identifizieren lassen. Die Ansicht „vw_ctrx_archive_session_perf" enthält allerdings nur die Gesamtheit der Prozessorlast einzelner Sitzungen pro Zeiteinheit und lässt keine Rückschlüsse darauf zu, welcher Natur die Last ist, d. h. welche Anwendungen die Benutzer im Einzelnen starten und wie hoch deren Ressourcenbedarf jeweils ist. Eine solche Auswertung gelingt erst in Verbindung mit der Ansicht „vw_image_perf". Letztere erfasst Leistungsdaten zu Instanzen von ausführbaren Dateien und ordnet diese den Sessions zu.

Für die Untersuchung hinderlich ist, dass EdgeSight keine Möglichkeit bietet, zweifelsfrei zu unterscheiden, ob es sich bei einem Prozess um eine vom Benutzer vordergründig genutzte Applikation (z. B. Microsoft Outlook, PowerPoint oder Word) oder um einen Systemprozess (z. B. das für jede Sitzung erforderliche Client/Server Run-Time Subsystem (csrss.exe) oder den VirenScanner) handelt. Weiterhin ist ohne Weiteres keine Klassifizierung der Applikationen nach ihrer Komplexität bezüglich der grafischen Oberfläche, der Interaktionsmöglichkeiten des Endanwenders und der daraus resultierenden Anzahl an I/O-Operationen und Prozessorlast möglich. Handelt es sich beispielsweise um eine Anwendung, wie den Texteditor notepad.exe, der mit einfachem GUI nur ebenso einfache Aktionen anbietet, um eine Anwendung mittlerer Komplexität, wie einen Browser oder die Textverarbeitung Microsoft Word, oder um eine komplexe Anwendung, wie z. B. OriginLab Origin, das bezüglich der damit möglichen Interaktionen und Berechnungen hohe Last hervorrufen kann?

Da diese Informationen für folgende Abfragen nützlich sind, wird zunächst eine Erweiterung der Datenbank eingeführt, die die fehlenden Informationen ergänzt. Die nachstehende Abfrage (Tabelle 12-12) ermittelt dazu zunächst alle in der EdgeSight-Datenbank bekannten ausführbaren Dateien, deren Beschreibung sowie den Hersteller.

Tabelle 12-12: Abfrage nach ausführbaren Dateien

```
SELECT
      imid,
      filename AS "Filename",
      description AS "Description",
      name AS "Vendor"
FROM
      image INNER JOIN
      vendor ON image.vendid = vendor.vendid
```

Quelle: Eigene Darstellung

Die resultierende Aufstellung (auszugsweise dargestellt in Tabelle 12-13) ist um die neuen Spalten „image_type" sowie „app_class" zu ergänzen, in denen durch einen Systemadministrator der Terminal Server für jede ausführbare Datei zu qualifizieren ist, ob es sich zum einen um eine Anwendung mit direktem Nutzen für die Benutzer („app") oder um einen Systemprozess („system") handelt. Zum anderen sind die ausführbaren Dateien ihrer Komplexität nach zu klassifizieren. Die Systemprozesse treten für die Endanwender in der

Regel nicht in Erscheinung, sondern laufen im Hintergrund („background"). Die Anwendungen werden nach den Klassen „simple", „medium" sowie „complex" unterschieden.

Tabelle 12-13: Aufstellung aller ausführbaren Dateien

imid	Filename	Description	Vendor	image_type	app_class
(…)	(…)	(…)	(…)	(…)	(…)
2	Armsvc.exe	Adobe Acrobat Update Service	Adobe Systems Inc.	system	background
13	Csrss.exe	Client Server Runtime Process	Microsoft Corporation	system	background
14	Ctxsvchost.exe	Citrix Service Host	Citrix Systems, Inc.	system	background
172	Excel.exe	Microsoft Excel	Microsoft Corporation	app	complex
182	Iexplore.exe	Internet Explorer	Microsoft Corporation	app	medium
190	Mcshield.exe	McShield.exe	McAfee, Inc.	system	background
204	Proquota.exe	ProQuota	Microsoft Corporation	system	background
318	Outlook.exe	Microsoft Office Outlook	Microsoft Corporation	app	medium
319	Snippingtool.exe	Snipping Tool	Microsoft Corporation	app	simple
325	Spinworks.exe	SpinWorks 3.1.8.1 (…)	Chemistry NMR Lab	app	complex
327	Origin9_64.exe	OriginLab Origin	Origin	app	complex
328	Citavi.exe	Citavi	Swiss Academic Software	app	medium
330	Chemdraw.exe	ChemDraw Pro 11.0	CambridgeSoft Corp.	app	medium
333	Icast.exe	ICA Start helper	Citrix Systems, Inc.	system	background
335	Splwow64.exe	Print driver host for 32bit appl.	Microsoft Corporation	system	background
340	Mindmanager.exe	Mindjet MindManager 8	Mindjet	app	complex
343	Notepad++.exe	Notepad++	Don HO don.h@free.fr	app	simple
(…)	(…)	(…)	(…)	(…)	(…)

Quelle: Eigene Darstellung

Anschließend ist diese Aufstellung in zwei Tabellen zu überführen, wobei alle redundanten Informationen zu entfernen sind, so dass allein die Zuordnung der Image-ID zum jeweiligen Typ und der Klasse verbleibt (Tabelle 12-14). Diese Tabellen werden dann als kommaseparierte Listen (Comma Separated Values (CSV)) gespeichert.

Tabelle 12-14: Import-Vorlage für die Tabelle „image_type"

imid	image_type
(…)	(…)
2	system
13	system
14	system
172	app
182	app
190	system
204	system
318	app

imid	app_class
(…)	(…)
2	background
13	background
14	background
172	complex
182	medium
190	background
204	background
318	medium

imid	image_type
319	app
325	app
327	app
328	app
330	app
333	system
335	system
340	app
343	app
(…)	(…)

imid	app_class
319	simple
325	complex
327	complex
328	medium
330	medium
333	background
335	background
340	complex
343	simple
(…)	(…)

Quelle: Eigene Darstellung

Werden daraufhin innerhalb der EdgeSight-Datenbank die neuen Tabellen „image_type" und „image_app_class" angelegt und mit dem Inhalt der CSV-Datei befüllt, so stehen die gewünschten Informationen zur Natur der Prozesse für weitergehende Abfragen zur Verfügung. (Hinweis: Das zuvor beschriebene Verfahren ist im Laufe weiterer Auswertungen einer produktiven EdgeSight-Datenbank periodisch zu wiederholen, um auch neu hinzukommende ausführbare Dateien zu erfassen.)

12.5.4.4 Prozessorlast und I/O-Operationen

Entsprechend kann nun mittels einer weiteren Abfrage (Tabelle 12-15) pro Zeiteinheit und Sitzung der Ressourcenbedarf unterteilt nach Applikationen und Systemprozessen ermittelt werden.

Das betrifft zum einen die Prozessorlast. Weiterhin liefert die EdgeSight-Datenbank auch Informationen zu I/O-Operationen. Konkret handelt es sich dabei um Informationen, die auch über den Windows-eigenen Performance Monitor abfragbar sind, darunter die Variablen „IO Data Bytes/sec", „IO Data Operations/sec", „IO Other Bytes/sec" sowie „IO Other Operations/sec". Die „…Data…" Variablen beinhalten die Operationen mit direktem Bezug zur Datenverarbeitung, die „…Other…" Variablen dagegen die mittelbar zur Datenverarbeitung erforderlichen Kontrolloperationen[125].

Diese Informationen werden in der Datenbank mit den Spalten „…Operations/sec" jeweils nach der Anzahl der Input/Output-Operationen pro Sekunde (IOPS) sowie mit den Spalten „…Bytes/sec" nach der Datenübertragungsrate erfasst. Die Datenübertragungsrate umfasst dabei gleichermaßen Lese- sowie Schreiboperationen und schließt Netzwerk, Dateisystem sowie andere Geräte mit ein. Die Innensicht der virtuellen Maschinen, die EdgeSight erfasst, lässt – auch auf Grund der in Kapitel 12.1 beschriebenen Zusammenhänge von physischen und virtuellen Komponenten – allerdings keinen eindeutigen Rückschluss darauf zu, welche

[125] Informationen zum Processor Object: https://msdn.microsoft.com/en-us/library/ms804621.aspx (abgerufen am 01.02.2016).

tatsächliche Auslastung von physischen Festplattensubsystemen und Netzwerkkomponenten aus Abläufen innerhalb der VM resultieren. Daher werden bezüglich der Datenübertragungsraten die Messwerte aus den anderen Monitoringsystemen herangezogen.

Tabelle 12-15: Abfrage nach Ressourcenbedarfs pro Session

```
SELECT
      CONVERT(varchar(19), DATEADD(mi, timezone.daylight_bias,
           vw_image_perf.dtperiod), 120) AS "Timestamp",
      session_id AS "Session ID",
      image_type.image_type AS "Type",
      SUM(vw_image_perf.total_processor_time_sum /
           vw_image_perf.total_processor_time_cnt) AS "CPU Load",
      SUM(vw_image_perf.data_operations_sec_sum /
           vw_image_perf.data_operations_sec_cnt) AS "Data Ops",
      SUM(vw_image_perf.other_operations_sec_sum /
           vw_image_perf.other_operations_sec_cnt) AS "Other Ops"
FROM
      vw_image_perf INNER JOIN
      instance ON vw_image_perf.instid = instance.instid INNER JOIN
      image_type ON vw_image_perf.imid = image_type.imid INNER JOIN
      timezone ON instance.tzid = timezone.tzid
WHERE
      DATEPART(…)
      DATEPART(…)
      DATEPART(…)
      CONVERT(…)
      CONVERT(…)
GROUP BY
      CONVERT(varchar(19), DATEADD(mi, timezone.daylight_bias,
           vw_image_perf.dtperiod), 120),
      session_id,
      image_type
ORDER BY
      "Timestamp", session_id
```

Quelle: Eigene Darstellung

Die zuvor beschriebene Auswertung des Ressourcenbedarfs ist mit einer weiteren Abfrage (Tabelle 12-16) auf Grund der Erweiterung der Datenbank ebenso im Hinblick auf die Anwendungsklassen möglich.

Tabelle 12-16: Abfrage von CPU-Last und I/O nach „app_class“

```
SELECT
      image_app_class.app_class AS "Class",
      COUNT(*) AS "Count",
      AVG(vw_image_perf.data_operations_sec_sum /
            vw_image_perf.data_operations_sec_cnt) +
            AVG(vw_image_perf.other_operations_sec_sum /
            vw_image_perf.other_operations_sec_cnt) AS "Total Ops",
      AVG(vw_image_perf.data_bytes_sec_sum /
            vw_image_perf.data_bytes_sec_cnt) +
            AVG(vw_image_perf.other_bytes_sec_sum /
            vw_image_perf.other_bytes_sec_cnt) AS "Total Bytes"
FROM
      vw_image_perf INNER JOIN
      instance ON vw_image_perf.instid = instance.instid INNER JOIN
      image ON vw_image_perf.imid = image.imid INNER JOIN
      image_type ON vw_image_perf.imid = image_type.imid INNER JOIN
      image_app_class ON vw_image_perf.imid = image_app_class.imid INNER JOIN
      machine ON instance.machid = machine.machid INNER JOIN
      timezone ON instance.tzid = timezone.tzid
WHERE
      DATEPART(…)
      DATEPART(…)
      DATEPART(…)
      CONVERT(…)
      CONVERT(…)
GROUP BY
      app_class
ORDER BY
      "Class"
```

Quelle: Eigene Darstellung

Die zuvor beschriebenen Ad-hoc Abfragen werten die EdgeSight-Datenbank im Hinblick auf die in Kapitel 6.5 erforderlichen Daten zur Parametrisierung des in der Software VITO entworfenen Modells aus. Im vorherigen Appendix A sind die stochastischen Modelle und Algorithmen beschrieben, die der Modellierung in VITO zu Grunde liegen.

12.6 Performance-Daten der Citrix XenServer

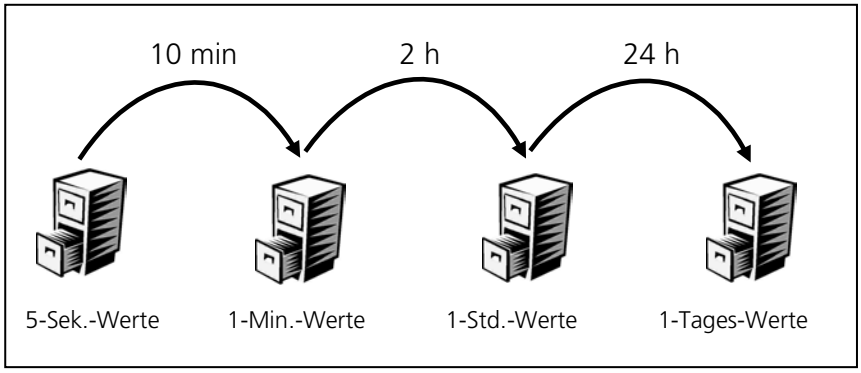
Über SQL-Abfragen lassen sich jedoch nur die Daten der XenApp Terminal Server ermitteln, also Daten, die innerhalb der virtuellen Maschinen erfasst worden sind. Um zu untersuchen, wie diese Daten mit den tatsächlichen Performance-Werten der Virtualisierungshosts korrelieren, sind weitere Informationen erforderlich, die vom Hypervisor abgefragt werden können. Unter XenServer 6.1 war dazu noch die Installation einer separaten Komponente, des

sogenannten Performance Monitoring Enhancements Pack[126], nötig. Ab XenServer 6.2 ist die entsprechende Funktion bereits im Betriebssystem enthalten[127].

Der XenServer erfasst Performance-Daten zum Hypervisor selbst und zu den darauf laufenden Maschinen in Round Robin Databases (RRD). Dabei handelt es sich um Dateien fester Größe, die für eine definierte Zeitspanne über die Zeit veränderliche Werte erfassen. Dazu wird jeweils der für die Zeitspanne nötige Speicherplatz alloziert. Die Datenbank wächst nicht darüber hinaus, sondern überschreibt nach dem Prinzip eines Ringpuffers die ältesten Werte, wenn das Ende eines Zeitinvervalls erreicht wird. Eine RRD wird in der Regel in mehrere Round Robin Archives (RRA) unterteilt, um mehrere Zeitspannen mit unterschiedlichem Detaillierungsgrad zu erfassen. Die RRD im XenServer sind unterteilt in vier Hauptsegmente mit folgender Detailtiefe (vgl. [HP, 2010], S. 3/4 und Abbildung 12-15):

- Für den Zeitraum der letzten zehn Minuten werden alle fünf Sekunden aktuelle Werte zu den jeweiligen Performance-Variablen erfasst.
- Für die letzten zwei Stunden werden die Werte aus dem vorherigen Abschnitt zu minütlichen Werten konsolidiert. Dabei werden Mittelwert, Minimum und Maximum gespeichert.
- Für die letzte Woche werden die Werte aus dem vorangegangenen Abschnitt zu stündlichen Werten konsolidiert. Auch hier werden Mittelwert, Minimum und Maximum gespeichert.
- Für ein Jahr werden die Werte aus dem vorherigen Abschnitt zu täglichen Werten konsolidiert. Wiederum werden Mittelwert, Minimum und Maximum gespeichert.

Abbildung 12-15: Datenkonsolidierung im XenServer



Quelle: Eigene Darstellung

Die RRD sind als Dateien im XPORT-Format jeweils im Dateisystem der XenServer Hosts im Pfad „/var/xapi/blobs/rrds" mit dem Namensschema „*UUID*.gz" gespeichert, wobei der Universally Unique Identifier (UUID) als Platzhalter für die eindeutige Identifikationsnummer des

[126] Performance Monitoring Enhancements Pack for XenServer 6.1: http://support.citrix.com/article/CTX135033 und Update http://support.citrix.com/article/CTX138750 (Links abgerufen am 01.02.2016).
[127] XenServer 6.2.0 Release Notes: http://support.citrix.com/article/CTX137826#newfeatures (abgerufen am 01.02.2016).

XenServers oder einer der VM steht. Die Daten werden vom XenServer auch per HTTP bereitgestellt. Ein Link der Form

```
http://root:<Passwort>@xen01/host_rrd
```

authentifiziert beispielsweise den Benutzer „root" mit seinem Passwort <*Passwort*> am Server „xen01" und ruft die RRD des Hypervisors ab. Analog liefert der folgende Befehl die statistischen Daten einer VM mit dem angegebenen UUID, die auf dem Hypervisor„xen01" läuft:

```
http://root:<Passwort>@xen01/vm_rrd?uuid=e4506347-a790-220d-5a1a-a122d2c709fd
```

Um die Daten zu extrahieren und für die weitere Auswertung zu konvertieren, wurde ein Linux-System benutzt, auf das die Daten von allen XenServer Hosts des Beispielszenarios heruntergeladen wurden. Auf diesem Linux-System wurde im ersten Schritt zunächst das Werkzeug „RRDtool"[128] in der Version 1.4.8 extrahiert, kompiliert und installiert:

```
tar -xvf rrdtool.tar.gz
cd rrdtool-1.4.8
./configure --disable-rrd_graph
make
make install
```

Das weitere Vorgehen ist im Folgenden exemplarisch für den Server „xen01" beschrieben. Der nachstehende Befehl lädt die RRD im XML-Format herunter:

```
wget -O xen01.xml http://root:<Passwort>@xen01/host_rrd
```

Anschließend sind in der XML-Datei alle Vorkommen des Strings „Infinity" zu ersetzen:

```
sed -i 's/Infinity/1000000000000000/g' xen01.xml
```

Dann konvertiert RRDtool die XML-Datei wieder in eine RRD:

```
rrdtool restore xen01.xml xen01.rrd
```

Aus der resultierenden RRD-Datei können anschließend ebenfalls mittels RRDtool die statistischen Daten für wählbare Zeiträume abgefragt und als kommaseparierte Liste (Comma Separated Values (CSV)) gespeichert werden. Dabei markiert der Parameter „-s" den Startzeitpunkt und der Parameter „-e" den Endzeitpunkt der Abfrage:

```
rrdtool fetch xen01.rrd AVERAGE -r 5 -s -1week -e -1hour > xen01.csv
```

[128] RRDtool: http://oss.oetiker.ch/rrdtool/ (abgerufen am 01.02.2016).

Der Befehl zuvor verwendet allerdings relative Zeitangaben für den zu exportierenden Zeitraum, die vom bei der Abfrage aktuellen Zeitpunkt abhängen. Für eine spätere Auswertung gespeicherter RRD empfiehlt sich daher die Angabe absoluter Zeitpunkte, welche in diversen Formaten an RRDtool übergeben werden können[129], wie in folgendem Beispiel:

```
rrdtool fetch xen01.rrd AVERAGE -r 5 -s '12:30 July 21 2014' -e '13:30 July 21 2014'
> xen01.csv
```

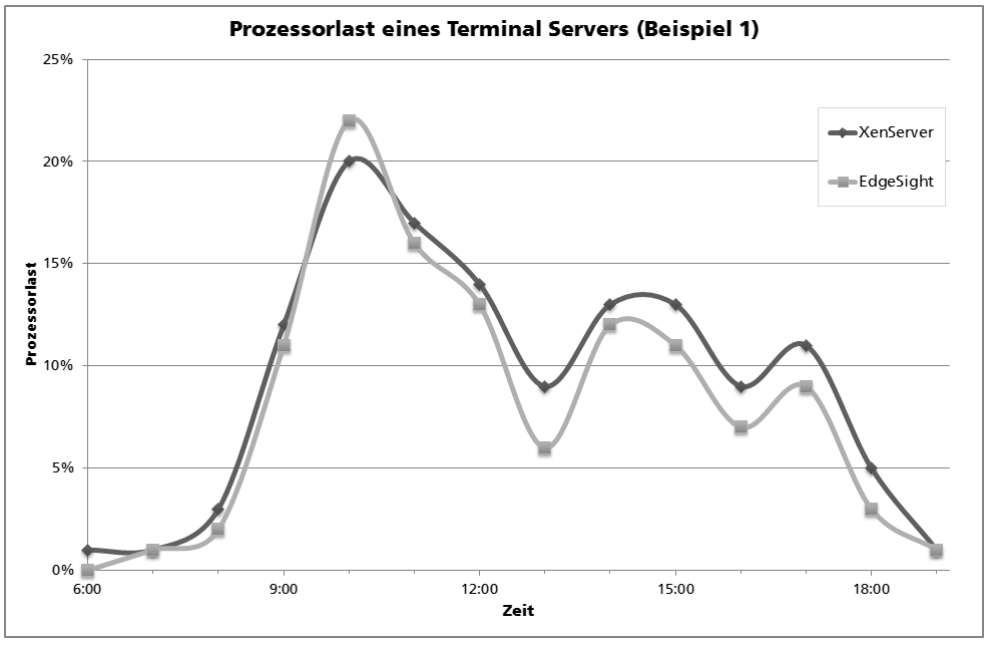
Für einen Import der CSV z. B. in Microsoft Excel sind als Trennzeichen sowohl Leerzeichen als auch „:“ anzugeben. Abschließend ist zu berücksichtigen, dass im resultierenden Datensatz die Zeitstempel als UNIX-Sekunden[130] angegeben sind. Die folgende Formel wandelt einen solchen Zeitstempel T in ein Datum nach dem Gregorianischen Kalender um, wobei „+2“ den Offset zur UTC repräsentiert, was in diesem Beispiel die Zeitzone der Mitteleuropäischen Sommerzeit (MEST) bezeichnet:

Formel 12-1

$$(((T/60)/60)/24) + DATUM(1970;1;1) + (+2/24) = (T/86400) + 25569 + (+2/24)$$

Die so gewonnenen Daten wurden für mehrere VM sowie mehrere Zeiträume im Hinblick auf die Prozessorlast ausgewertet und den Daten aus EdgeSight gegenüber gestellt.

Abbildung 12-16: Prozessorlast einer VM – Beispiel 1



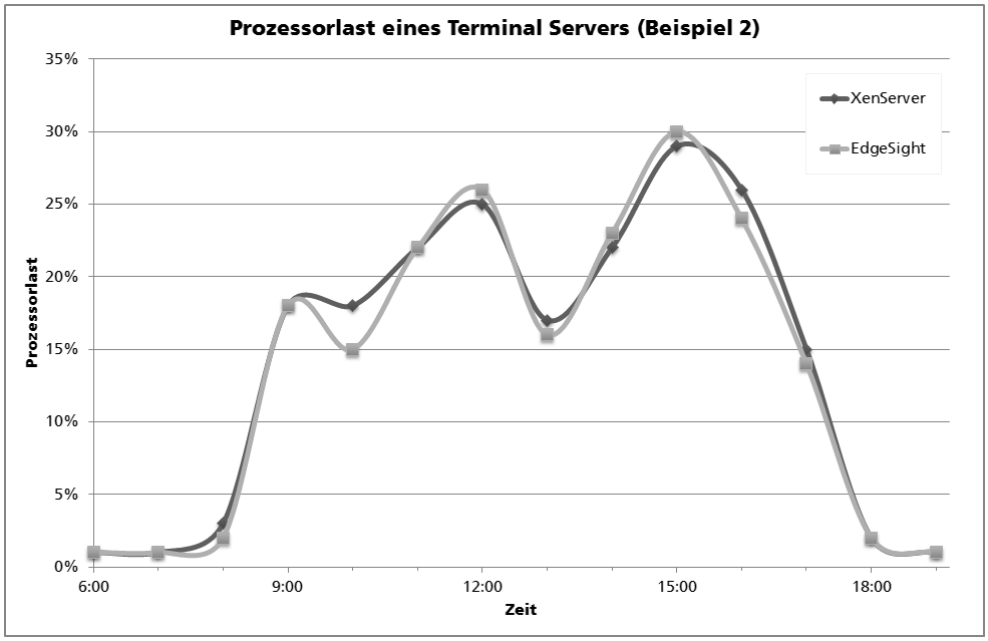
Quelle: Eigene Darstellung

[129] rrdfetch: http://oss.oetiker.ch/rrdtool/doc/rrdfetch.en.html#ITIME_REFERENCE_SPECIFICATION (abgerufen am 01.02.2016).
[130] Gemäß Definition der Unixzeit werden die vergangenen Sekunden seit Donnerstag, dem 01.01.1970 00:00 Uhr (UTC) gezählt.

Dazu wurden die separaten Daten für vier vCPU jeweils zu einem Gesamtwert aggregiert, da EdgeSight nur die gesamte Prozessorlast erfasst. Die Daten aus EdgeSight wurden mittels einer SQL-Abfrage gegen die Datenbank gewonnen (vgl. Tabelle 12-12).

Es zeigte sich dabei, dass die „Innensicht" des Windows-Betriebssystems bzw. EdgeSight-Agenten mit der „Außensicht" des XenServers auf die VM in engen Grenzen übereinstimmt (vgl. Abbildung 12-16 und Abbildung 12-17), weshalb im Falle der XenApp Worker für die weitere Verwendung und Parametrisierung des Modells die Daten aus EdgeSight herangezogen wurden, da auch sämtliche weitergehenden Informationen zu Sessions und veröffentlichten Anwendungen in EdgeSight verfügbar sind. Nur für die übrigen, nicht in EdgeSight erfassten Systeme, wie die Provisioning Server, wurden die Daten des XenServers verwendet.

Abbildung 12-17: Prozessorlast einer VM – Beispiel 2



Quelle: Eigene Darstellung

Die Performance-Werte des XenServers wurden nur benutzt, um die Auslastung der physischen Prozessoren der Virtualisierungshosts zu ermitteln. Für die Netzwerklast wurden demgegenüber die Werte aus dem LANrunner verwendet, da diese nur in diesem System als Werte vom Datentyp „Counter" vorliegen und somit die kumulierte übertragene Datenmenge über den Betrachtungszeitraum ausweisen.

12.7 Alternativen und weitere Entwicklung

Ebenso wie die Architektur der Terminal Server Lösung XenApp grundlegend weiterentwickelt wurde (vgl. Kapitel 2.3.2), wurde in diesem Zug auch das Monitoring komplett neu implementiert. Wenn das hier entwickelte Referenzmodell einer SBC Infrastruktur für das

Kapazitätsmanagement auf neuere Versionen der Software und auf das Konzept der Desktop-Virtualisierung erweitert werden soll, sind entsprechend alternative Wege zur Gewinnung der nötigen Daten zu wählen, um das Modell zu parametrisieren. Im Kontext von Citrix XenDesktop stehen Performance-Daten über eine neue Monitoring-Schnittstelle zur Verfügung, die das standardisierte, auf HTTP basierende Protokoll OData (Open Data Protocol[131]) verwendet. Sowohl das Schema[132] als auch die Methoden zur Abfrage der Daten sind dokumentiert und es gibt Handreichungen zur Auswertungen mittels individuell angepasster Berichte[133].

Als Alternative zum in XenApp und XenDesktop integrierten Monitoring kann zudem bei einer Weiterentwicklung der Methodik die Software „Splunk“[134] zum Einsatz kommen. Es handelt sich dabei um ein Werkzeug zur Auswertung großer Mengen von Log-Daten im Hinblick auf Monitoring und Report-Erstellung. Mit „uberAgent for Splunk“[135] existiert eine Erweiterung zu diesem System speziell für die Untersuchung von Benutzersitzungen in Terminal Server Umgebungen und auf virtuellen Desktops. Einen ähnlichen Ansatz verfolgt auch die Software-Lösung Lakeside SysTrack[136], welche es ebenso für Terminal Server und virtuelle Desktops unter Microsoft Windows Betriebssystemen erlaubt, Messwerte bezüglichen der tatsächlichen und vom Benutzer wahrgenommenen Leistungsfähigkeit eines Systems zu erfassen und auszuwerten. Mit den zuvor genannten Werkzeugen lassen sich auch über das im Kontext dieser Arbeit exemplarisch untersuchte Beispielszenario hinaus reine Microsoft Remote Desktop Umgebungen oder solche basierend auf den Lösungen anderer Drittanbieter, wie z. B. Citrix XenDesktop oder VMware Horizon, im Hinblick auf für die Modellierung relevante Performance-Werte abfragen.

Unabhängig vom zur entfernten Präsentation verwendeten System ist zudem eine weitergehende Untersuchung der Auslastung des Dateiservers von großem Interesse. Wie in Kapitel 12.1 erläutert war es im Rahmen des Beispielszenarios nicht ohne weiteres möglich, den am Dateiserver beobachteten Netzwerkverkehr eindeutig einzelnen VM oder gar den in den VM laufenden Prozessen zuzuordnen. Erschwerend kam hinzu, dass der Dateiserver im Beispielszenario nicht exklusiv von den Hosts und VM der SBC Infrastruktur sondern auch von weiteren Servern und Clients genutzt wurde. Hier kann mit NetFlow[137] eine ursprünglich von der Firma Cisco Systems entwickelte Technik zum Einsatz kommen. NetFlow exportiert tiefergehende Informationen zum IP-Netzwerkverkehr und macht, mit Zeitstempeln versehen, die Anzahl ausgetauschter Bytes und Pakete zwischen Quell- und Ziel-IP-Adressen sowie Quell- und Ziel-IP-Ports einer genaueren Analyse zugänglich.

131 Open Data Protocol (OData): http://www.odata.org (abgerufen am 01.02.2016).
132 http://blogs.citrix.com/2013/08/27/xendesktop-7-monitor-service-what-data-is-available/ (abgerufen am 01.02.2016).
133 Citrix Director – Analyzing the Monitoring Data by Means of Custom Reports: http://blogs.citrix.com/2015/02/12/citrix-director-analyzing-the-monitoring-data-by-means-of-custom-reports/ (abgerufen am 01.02.2016).
134 Splunk: http://www.splunk.com (abgerufen am 01.02.2016).
135 uberAgent for Splunk: http://helgeklein.com/uberagent-for-splunk/ (abgerufen am 01.02.2016).
136 Lakeside SysTrack: http://lakesidesoftware.com (abgerufen am 01.02.2016).
137 NetFlow wurde von Cisco Systems entwickelt, jedoch mit der Version 9 in einem RFC (http://tools.ietf.org/html/rfc3954 (abgerufen am 01.02.2016)) als offener Standard definiert und von vielen weiteren Herstellern übernommen.

Die Untersuchung der zuvor genannten Technologien wird Gegenstand von auf diesem Dokument aufbauenden Folgearbeiten (vgl. Kapitel 7.2) sein mit dem Ziel, das in Kapitel 5.4 entworfene Referenzmodell auch auf zukünftige Infrastrukturen anzuwenden und entsprechend parametrisieren zu können.