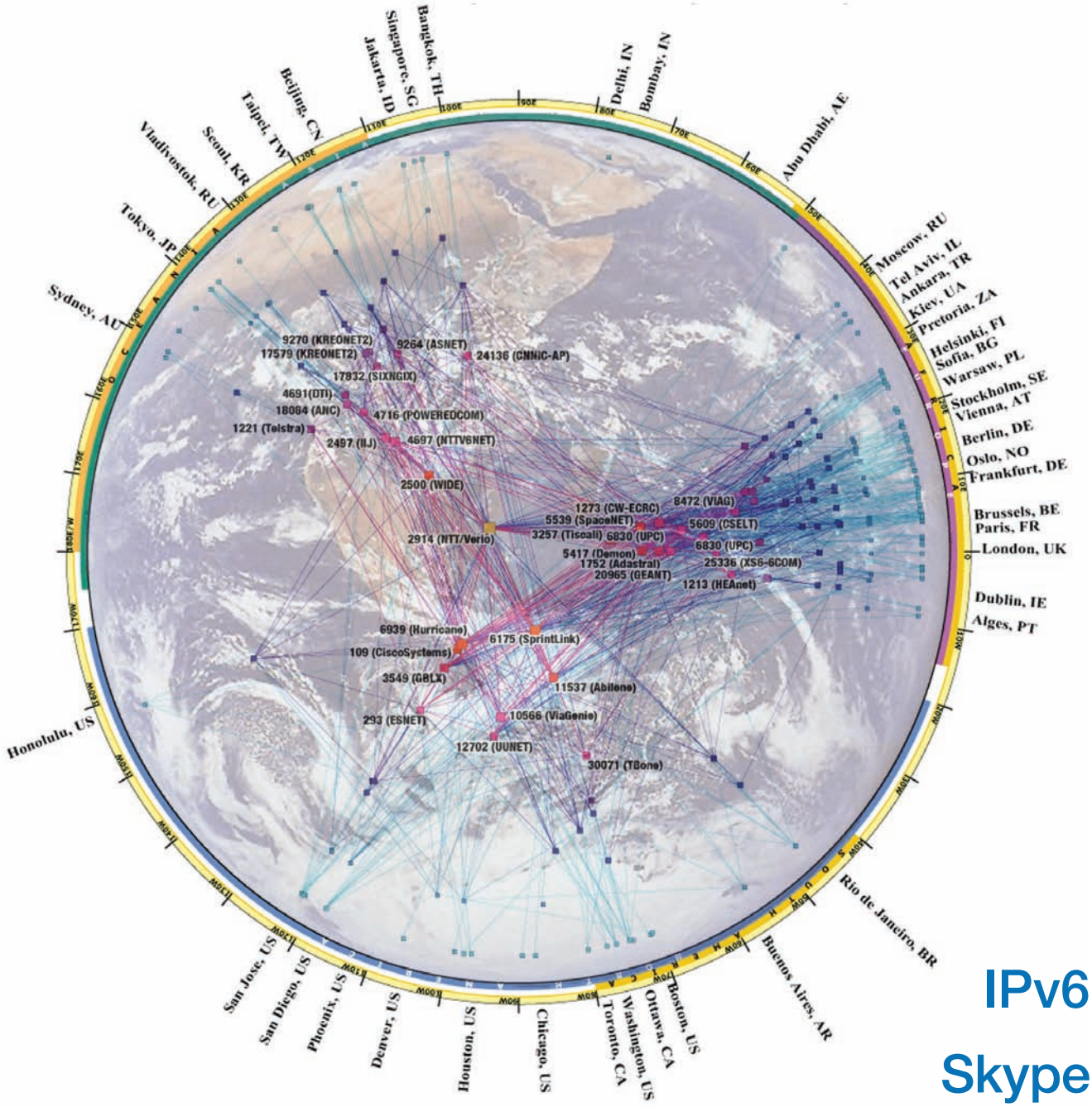
Nr. 15 / Dezember 2006

ISSN 1605-475X

ZID line

INFORMATIONEN DES ZENTRALEN INFORMATIKDIENSTES DER TU WIEN



IPv6

Skype

GemStone/S

Inhalt

Impressum / Offenlegung gemäß § 25 Mediengesetz:

Herausgeber, Medieninhaber:
Zentraler Informatikdienst
der Technischen Universität Wien
ISSN 1605-475X

Grundlegende Richtung: Mitteilungen des Zentralen Informatikdienstes der Technischen Universität Wien

Redaktion: Irmgard Husinsky

Adresse: Technische Universität Wien,
Wiedner Hauptstraße 8-10, 1040 Wien
Tel.: (01) 58801-42014, 42002
Fax: (01) 58801-42099
E-Mail: zidline@zid.tuwien.ac.at
www.zid.tuwien.ac.at/zidline/

Erstellt mit Corel Ventura
Druck: Grafisches Zentrum an der TU Wien,
1040 Wien, Tel.: (01) 5863316

Editorial

Das neue Internet Protocol Version 6 (IPv6) ermöglicht eine gewaltige Erweiterung des Adressraums – damit wir auch den Kühlschrank ans Internet anschließen können. An der TU Wien ist schon alles dafür vorbereitet.

Es gibt eine große Anzahl von Linux-Distributionen, sodass sich die Frage stellt: „Linux, aber welches?“ Lesen Sie dazu einige Gedanken in dieser ZIDline.

Ferner liefern wir einen Beitrag zur Diskussion über kostenloses Telefonieren im Internet (Stichwort: Skype).

Die TU-Website wird zurzeit auf das Content Management System TYPO3 umgestellt. Der Relaunch wird noch im Dezember erfolgen.

Für Berechnungen aus dem Gebiet der Finiten Elemente und der Strömungsdynamik gibt es seit vielen Jahren zentrale leistungsfähige Rechner an der TU. Der neueste ist der IBM Power5+ Cluster.

Kollege Mikulka stellt GemStone/S und die objektorientierte Datenbank vor, die seit Jahren am ZID im Einsatz ist.

Ein Honeynet lockt Angreifer im Netz an. Somit können Informationen gesammelt werden, ohne dass jemand geschädigt wird. Lesen Sie den Beitrag über das Honeynet-Projekt auf Seite 24.

Das Rechnerlabor für Virtuelle Produktentwicklung wurde auch mit Campuslizenzen für CAx- und PDM-Software ausgerüstet.

Vielen herzlichen Dank allen Autoren dieses Hefts für ihre Kooperationsbereitschaft, sowie allen, die an der Fertigstellung dieser ZIDline mitgearbeitet haben.

Mit den besten Wünschen für ein erfolgreiches Jahr 2007.

Irmgard Husinsky

Titelbild-Bearbeitung: A. Klauda

Die ZIDline im Web:

www.zid.tuwien.ac.at/zidline/

IPv6 – im TUNET kein Problem

Johann Kainrath

Mit IPv4 sind wir mittlerweile bestens vertraut und es ist aus unserem täglichen Leben kaum mehr wegzudenken. Welche IP-Adresse hat Ihr Rechner? Können Sie das Gateway pingen? Verwenden Sie einen VPN-Client mit IPSec? Solche oder ähnliche Fragen müssen wir uns häufig stellen lassen. Wird das mit IPv6 anders oder besser sein? Nun, die Entwickler von IPv6 haben aus mehr als 15 Jahren mit IPv4 gelernt und einiges einzubauen versucht, das uns das tägliche Leben erleichtern wird. Ist es real, dass alle unsere Geräte (vom PDA über Haushaltsgeräte bis zum Auto) in Zukunft via Internet erreichbar sein sollen?

IPv4, Grundlage für das Internet

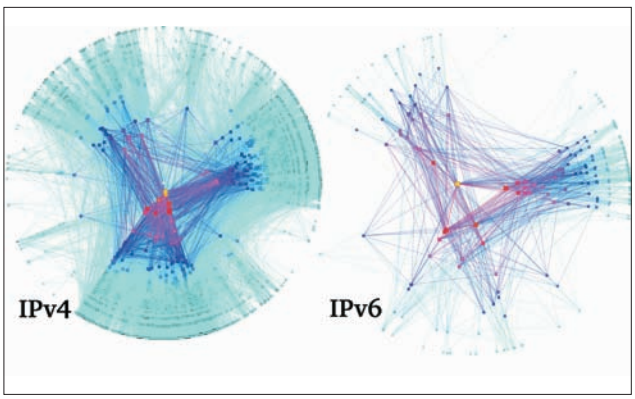
Basisübertragungsprotokoll im Internet ist derzeit das Internet Protocol Version 4 (IPv4). Damit man mit anderen Netzteilnehmern lokal wie global kommunizieren kann, benötigt jedes Gerät mit Internet-Anschluss eine weltweit eindeutige IP-Adresse. Eine IPv4-Adresse (z. B. 128.130.2.3) hat eine Länge von 32 Bit, das ergibt 2^{32} (rund 4,3 Milliarden) mögliche Adressen, von denen allerdings aus technologischen Gründen nicht alle für Endgeräte verwendet werden können.

Der IPv4-Adressraum wäre schon längst zu klein, gäbe es nicht diverse Hilfsmittel. NAT (Network Address Translation, auch bekannt unter dem Namen Masquerading) in Zusammenhang mit privaten Adressen ermöglicht mit nur einer IP-Adresse mehreren Endgeräten den Zugang zum Internet.

Das rasante Wachstum und die globale Verbreitung des Internet war zum Zeitpunkt der Konzeptionierung von IPv4 (1981) noch nicht absehbar. Aus diesen Gründen wurde bereits 1995 mit der Entwicklung des Internet-Protokolls der nächsten Generation (IPv6) begonnen.

Die TU Wien nahm bereits sehr früh am Internet teil und am 18. Juni 1986 wurden der TU Wien die noch heute verwendeten Class B Adressen vom USC/Information Sciences Institut zugeteilt. Die beiden vom damaligen regionalen Internet-Registrar zugeteilten IPv4 Class B Adressblöcke umfassten die Netze 128.130.0.0/16 und 128.131.0.0 (je 65.536 Adressen) sowie einige kleinere Class C Netzbereiche mit je 256 Adressen. Bis heute konnte damit an der TU Wien das Auslangen gefunden werden, auch wenn es in bestimmten Bereichen des externen Zugangs (TU-ADSL, VPN, Dialin, WLAN) zunehmend eng wird.

Technisch gesehen besteht derzeit kein unmittelbarer Grund, zu IPv6 zu migrieren. NAT wird oft auch als Security Feature verwendet, ist aber dafür nicht wirklich designed und geeignet. Viele neue Internet-Applikationen wie VoIP, Peer-to-Peer etc., die sich IPv4 als Transport bedienen, haben damit ein Problem. NAT verhindert eine echte End-zu-End-Konnektivität und damit Transparenz. Verbindungsaufbauten sind quasi nur von innen möglich.



Visualisierung IPv4/IPv6 Adressraum, 2005
Cooperative Association for Internet Data Analysis (CAIDA)
www.caida.org/analysis/topology/as_core_network/ipv6.xml

IPv6, das neue Internet-Protokoll

Größerer Adressraum

Der wichtigste und offensichtlichste Vorteil von IPv6 (RFC 2460) ist, dass die Adressen länger sind, daher Platz für einen viel, viel größeren Adressraum ist. Die tatsächliche Anzahl individueller Adressen, die mit den nun zur

Verfügung stehenden 128 Bit möglich sind, geht weit darüber hinaus, was sich jemand, der kein Astronom oder Teilchenphysiker ist, vorstellen kann:

340.282.366.920.938.463.463.374.607.431.768.211.456

Das sind 2^{128} IP-Adressen. Dagegen wirkt die Anzahl der möglichen IPv4-Adressen (2^{32}) eher banal:

4.294.967.296

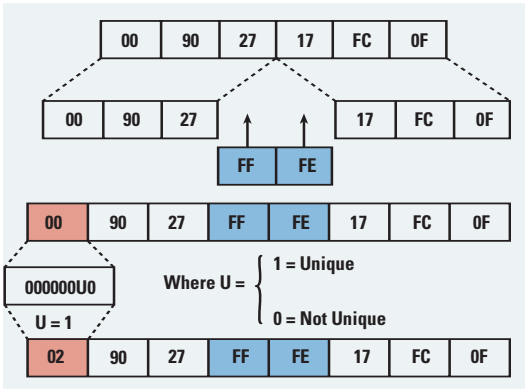
Der 128-Bit Adressraum ist groß genug, um ca. 667 Billiarden Adressen pro Quadratmillimeter Erdoberfläche zur Verfügung zu stellen (ca. $6{,}5*10^{28}$ Adressen pro Mensch). Angenommen, der benötigte Adressraum würde sich alle fünf Jahre verdoppeln, so würde der IPv6 Adressraum bis zum Jahr 2485 ausreichen.

Die Befürchtung, dass die IPv4-Adressen ausgehen, hat sich noch nicht bewahrheitet, da die Adressvergabe nicht mehr exponentiell ansteigt. Bis zur flächendeckenden Verwendung von IPv6 stehen noch ausreichend Adressen zur Verfügung.

Weitere Verbesserungen neben der Ausweitung des Adressraumes sind *Mobile IP* und vereinfachte Umnummerierung (*Renumbering),* Security (IPSec), *Quality of Service* (QoS) und Multicast „serienmäßig“ sowie Effizienz bei *Routing* und *Packet Processing*. Die automatische Adressvergabe an Endsysteme in Form der *Stateless Autoconfiguration* stellt einen wesentlichen Vorteil gegenüber IPv4 dar.

Stateless Autoconfiguration, Plug&Play

Die Zuweisung von IPv6-Adressen an die Arbeitsplatzrechner erfolgt standardmäßig mittels *Stateless Autoconfiguration*. Diese Methode wird inzwischen von allen gängigen Betriebssystemen unterstützt und hat den Vorteil, dass die Vergabe sehr einfach und ohne manuelle Konfigurationsänderungen des PCs funktioniert (Plug & Play). Die auf solche Weise vergebenen IPv6-Adressen werden nach der Norm EUI-64 gebildet und enthalten daher auch die MAC-Adresse des jeweiligen Rechners. (Der EUI-64-Name der MAC-Adresse bildet bei der IPv6-Autokonfiguration in der Regel die letzten 64 Bit der IPv6-Adresse.)



IPv6 Adressierung, EUI-64

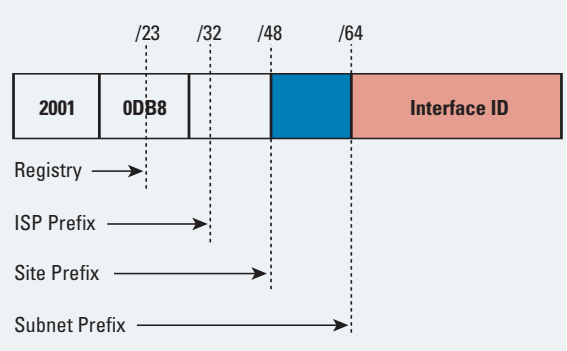
Bei der *stateless* Autokonfiguration bekommt ein IPv6-System in der Regel vom zuständigen Router vollautomatisch einen Präfix (mithilfe dessen die IP-Adresse gebildet wird) und ein Gateway zugewiesen (sowie einige weitere Parameter), nicht aber einen DNS-Server. Bei der Methode mit DHCPv6 (*stateful*) kann auch eine Nameserver IP-Adresse zugewiesen werden. Der spezielle Mechanismus DAD (*Duplicate Address Detection*) verhindert die doppelte Vergabe von IPv6-Adressen.

IPv6 Adressen

IPv6-Adressen werden nicht in dezimaler (zum Beispiel 128.131.192.13), sondern in hexadezimaler Notation mit Doppelpunkten geschrieben, die die Adresse in acht Blöcke mit einer Länge von jeweils 16 Bit unterteilen. Beispiel einer IPv6-Adresse:

2001:629:400:36a:20c:29ff:fele:5ac8/64

Eine oder mehrere 16-Bit-Gruppen mit dem Wert 0000 können durch zwei aufeinander folgende Doppelpunkte ersetzt werden. Die resultierende Adresse darf höchstens einmal zwei aufeinander folgende Doppelpunkte enthalten. 2001:0629::fece:7c61 ist gleichbedeutend mit 2001:0629:0000:0000:0000:0000:fece:7c61, aber 2001::fece::7c61 ist nicht korrekt, da nicht nachvollzogen werden kann, wie viele 16-Bit-Gruppen durch die zwei Doppelpunkte jeweils ersetzt wurden. Führende Nullen einer 16-Bit-Gruppe dürfen ausgelassen werden, 2001:629::26:c ist gleichbedeutend mit 2001:0629::0026:000c.



Address Allocation and Assignment

Adressbereiche werden bei IPv6 durch Präfixe angegeben. Dazu wird die Präfixlänge (Anzahl der „gültigen“ Bits) als Dezimalzahl mit vorangehendem „/“ an die IPv6-Adresse angehängt. Subnetze werden als Adressbereiche ebenfalls durch den Präfix bestimmt. Die ersten 64 Bit der IPv6-Adresse dienen üblicherweise der Netzadressierung, die letzten 64 Bit werden zur Host-Adressierung verwendet. Beispiel: hat ein Netzwerkgerät die IPv6-Adresse 2001:629:400:36a:20c:29ff:fele:5ac8/64 so stammt es aus dem Subnetz 2001:629:400:36a::/64, das mit den ersten 64 Bit seiner Adresse identifiziert wird. Analog gehört das Subnetz, 2001:629:400:36a::/64 hierarchisch zum Subnetz mit dem kürzeren Präfix 2001:629:400::/48.

Die *Global Unicast Address*, die bei der *End-to-end* Kommunikation ins Internet verwendet wird, ist somit eine Kombination von Präfix und Interface-ID. Neben dieser global gültigen Adresse gibt es auch noch eine so genannte *Link-Local Address* (beginnend mit fe80::/10), die jedes IPv6-System automatisch generiert. Diese ist nur lokal im VLAN gültig. Datenpakete mit solchen Adressen werden vom zuständigen Router nicht weiter geleitet.

Type	Binary	Hex
Aggregatable Global Unicast	0010 0001	2001::/16
Link-Local Unicast	1111 1110 10	FE80::/10
Unique Local Unicast	1111 1100 1111 1101	FC00::/8 FD00::/8
Multicast	1111 1111	FF00::/16

Die IPv6-Adressen folgen einem hierarchischen Adress- und Aggregationsmodell

IPv6 und DNS

Wenn man von IPv6 DNS (*Domain Name Service*) spricht, gibt es zwei unterschiedliche Aspekte, die beachtet werden müssen. Einer davon ist, ob der betroffene Nameserver überhaupt Quad-A (AAAA) Records unterstützt, d. h. einen Namen in eine IPv6-Adresse auflösen kann. Der andere Gesichtspunkt bezieht sich darauf, ob der auf dem Klienten installierte Resolver (d. h. das Programmstück, das die Nameserverauflösung für die Internet-Applikationen vornimmt) IPv4 oder IPv6 als Transport verwendet, wenn es so genannte Lookups vornimmt. Für den Lookup von IPv6-Records muss der Resolver des verwendeten Betriebssystems dies unterstützen. Das neue VISTA-Betriebssystem von Microsoft unterstützt jede Variante, auch den reinen IPv6-Betrieb, sowohl die Namensauflösung als auch die wirkliche Kommunikation (also der Datenaustausch) finden über IPv6 statt.

IPv6 an der TU

Bereits Ende Februar 2001 erfolgten erste Modifikationen an der TUNET-Datenbank, um neue IPv6 DNS Attribute und damit IPv6 DNS Resource Records zu unterstützen. Im Mai 2001 erfolgte die Installation eines eigenen Routers, um Native IPv6 Connectivity zu bieten. Damit konnten erste Gehversuche unternommen werden. Am 26. 6. 2002 ging schließlich der für den Echtbetrieb geplante Cisco 2621 IPv6 Router in Produktion und realisierte die Anbindung. Bisher haben einige Institute diese Connectivity im Rahmen von Projekten genutzt.

Im Mai 2006 wurde ein neues erweitertes IPv6-Netz für die TU Wien (2001:629::/32) bei RIPE registriert (2^{96} IP-Adressen, gemäß RFC 3177). Für die Organisationseinheiten der TU werden Adressbereiche mit einem /48 Präfix (2001:629::/48) zur Verfügung gestellt, das sind 2^{80} Adressen. In zahlreichen Umbauten und Software-Upgrades wurde das TUNET *IPv6 ready* gemacht. Dank Dual-Stack (Koexistenz von IPv4 und IPv6 auf einer Netzwerkschnittstelle) ist es im TUNET möglich, IPv6 ohne zusätzliche Hardware (im selben VLAN wie IPv4) zu verwenden.

Wie kommt man zu IPv6-Adressen bzw. einem Adressbereich im TUNET? Auf Wunsch des jeweiligen Instituts kann IPv6 in den Instituts-Subnetzen aktiviert werden (E-Mail an hostmaster@noc.tuwien.ac.at). Für Server ist es sinnvoll, fixe Adressen zu verwenden, diese werden vom ZID direkt in das DNS eingetragen (kein EUI-64). Es ist darauf zu achten, dass entsprechende Security-Maßnahmen getroffen werden (Firewall, ...). Bei Arbeitsplatzrechnern kann der Vorteil der Autokonfiguration ausgenutzt werden (jedoch kein dynamic DNS). Aus Gründen des hohen administrativen Aufwands ist vorerst kein manueller DNS-Eintrag durch den ZID geplant.

Da derzeit das primäre Nameservice über IPv4 läuft, ist die automatische Konfiguration mit IPv6 Nameservern noch nicht notwendig. An einer Erweiterung des Standards wird gearbeitet, um zusätzliche Services wie NTP, DNS per Autokonfiguration den Clients zur Verfügung zu stellen.

Wie erwähnt, können die Nameserver der TU Wien derzeit wegen davorgeschalteter Firewall-Systeme zwar IPv6-Adressen (AAAA) auflösen, sind aber nicht direkt unter IPv6 erreichbar. Bis zu dem Zeitpunkt, wo diese Firewalls auch IPv6 unterstützen, erfolgt die DNS-Abfrage über das IPv4-Protokoll.

Wird IPv6 irgendwann IPv4 ersetzen?

Mittelfristig nein, langfristig eher ja. Die Dual-Stack Technologie ermöglicht den sanften Übergang, wobei *IPv6-only* Netze in nächster Zeit durchaus denkbar sind. Es scheint aber kein schlagartiges „Umschalten“ des Internets auf IPv6 (an einem so genannten D-Day) geplant zu sein. Tunnellösungen scheinen nur temporäre Lösungen auf dem Weg zu einer großflächigen Migration zu sein. Eigentlich ist kein aktuelles Betriebssystem für Endsysteme bekannt, welches IPv6 nicht unterstützt. Auch die wichtigsten Applikationen scheinen schon längere Zeit IPv6-fähig. Im Bereich der Netzwerkinfrastrukturgeräte (Router und Switches) ist IPv6 nicht wirklich ein Thema mehr und teilweise bereits in Hardware implementiert (zumindest aber in Software). Im Bereich der Firewall-Hersteller wird jedoch noch heftig an Lösungen gearbeitet, die an den Funktionsumfang von IPv4 Security-Gateways herankommen sollen.

Die Netzwerke sind zwar größtenteils *IPv6 ready*, aber die Treiber bzw. die Killerapplikationen sind noch nicht vorhanden. Insider und Experten, die internationale Meetings bereisen, sehen die Zukunft von IPv6 im IP/TV Servicebereich im südostasiatischen Raum bzw. in der Internet-Telefonie.

Tatsache ist, dass das Internet über IPv6 immer näher zum Benutzer kommt. War IPv6 bei Windows XP zwar bereits vorhanden, aber nicht sichtbar, so ist es bei vielen Unix-Systemen, bei MacOS X und nun auch bei VISTA bereits aktiv und in den Netzwerkeinstellungen sichtbar. Daher sind alle eingeladen, diese neue Technologie zu erkunden. Beispielsweise ist der Webserver des ZID (www.zid.tuwien.ac.at) ab sofort per IPv6 erreichbar.

Literatur und Links:

Iljitsch van Beijnum: Running IPv6, Apress, 2006, ISBN: 1-59059-527-0

Die Österreichische IPv6 Taskforce: www.austria.ipv6tf.org

IPv6 Dissemination and Exploitation: www.6diss.org

Ripe NCC Internet Resources: www.ripe.net/ipv6/

European IPv6 Internet Exchanges Backbone: www.euro6ix.org

Alte Projekte: www.6net.org, www.6bone.net

Linux, aber welches?

Rudolf Ladner, Walter Selos, Paul Torzicky

Welches Linux? Welches Auto? Welche Programmiersprache? Welcher Editor? Welches Waschmittel? Alle diese Fragen haben eines gemeinsam: Es existiert keine eindeutige Antwort und alle diejenigen, die glauben, eine gefunden zu haben, verteidigen dieselbe eifrig, emotional und oft irrational.

Ein zeitgemäßes Herangehen an diese Frage, indem man eine Internet-Suchmaschine in Anspruch nimmt, bringt ebenfalls kein befriedigendes Ergebnis. Gibt man unter Google „Welches Linux?“ ein, erhält man eine Flut von Links, die sich alle mit dem Thema beschäftigen. Schon ein kurzer Blick auf diese Adressen zeigt den vielschichtigen Charakter der Fragestellung. Ein URL, der einen gewissen Überblick geben kann ist *distrowatch.com*. Dort kann man eine kurze Beschreibung zu vielen Distributionen erhalten und wird über aktuelle Neuentwicklungen informiert. Allein die Tatsache, dass es auf dieser Seite auch eine Top-100 Liste der Linuxdistributionen gibt, zeigt erneut die Problematik der Auswahl der optimalen Version.

Grundsätzlich sollte man bei der Frage „Welches Linux?“ nie den Endzweck aus den Augen verlieren. Unter der Devise „Zum Einschlagen eines Nagels ist die Diskussion über die Farbe des Hammerstiels irrelevant“ sollte man sich nicht zu sehr in Details verlieren. Ist die Fragestellung „Warum Linux?“ beantwortet, ist die Auswahl der Distribution oft nur mehr Geschmackssache, denn im Großen und Ganzen kann man sagen, dass jede Distribution ihre Vor- und Nachteile hat.

Erwähnenswert ist die Tatsache, dass viele Distributionen eine so genannte Live-CD anbieten. Dabei handelt es sich um ein lauffähiges Linux auf CD, das ohne Installation auf der Harddisk exekutierbar ist. Damit kann man vorneweg checken, ob die gewählte Distribution die vorhandene Hardware erkennt und damit die grundsätzliche Verwendbarkeit gegeben ist. Außerdem besteht damit die Möglichkeit, ohne Installation in einem „Touch and Feel“-Prozess Erfahrungen zu sammeln und nicht zuletzt auch zu checken, ob einem die Distribution einfach „sympathisch“ ist oder nicht.

In der Folge werden wir einige der gängigsten Distributionen kurz vorstellen. Klarerweise erheben wir dabei keinerlei Anspruch auf Vollständigkeit und entschuldigen uns bei den Lesern, deren Lieblingsversion von Linux wir nicht angeführt haben.

Suse, Red Hat, Mandriva

www.suse.com, www.redhat.com, www.mandriva.com

Diese drei Distributionen werden von kommerziellen Firmen angeboten. Sie zeichnen sich durch ausgereifte Installationsprozeduren aus, die Windows-ähnlich mittels grafischer Interfaces mit dem Benutzer kommunizieren und damit klarerweise auch eine oft lästige Bevormundung mit sich bringen. Alle drei Distributionen haben bei verschiedenen neueren Hardware-Konfigurationen gepunktet aber auch versagt. Das Softwaremanagement wird bei allen über den Red Hat Package Manager (RPM) abgewickelt, der ein relativ ausgereiftes und komfortables Werkzeug darstellt. Standardfunktionen von Webserver-Betrieb mit dazugehörigen Datenbanken über Mailservice bis hin zu Networkprinting lassen sich problemlos installieren, konfigurieren und betreiben. Auch als Desktop-Systeme sind alle drei Distributionen ohne Aufwand konfigurierbar und zufrieden stellend zu betreiben. Bei Suse hat der in früheren Versionen oft kritisierte Systemmanager yast bzw. yast2 in den letzten Versionen entscheidende Verbesserungen erfahren.

Nicht unerwähnt bleiben soll die Tatsache, dass vor allem für Red Hat und Suse ein breites Angebot kommerzieller Software in angepassten Paketen vorliegt. Auch spezielle Anwendungen sind oft nur für die kommerziellen Distributionen vorhanden. So gibt es für die Itanium-Prozessoren von HP unterstützte Linux-Implementierungen nur mit Red Hat (Suse soll auch dazukommen).

Bei kommerziellen Herstellern ist aber mit Unzulänglichkeiten zu rechnen, die bei nicht-kommerziellen Distributionen unbekannt sind. So sind bei Red Hat für Updates kostenpflichtige Keys anzufordern, deren Bereitstellung immer wieder umgestellt wird, was immer wieder zu lästigen Verzögerungen führt.

Zusammenfassend kann man sagen, dass es sich um stabile und relativ gut durchstrukturierte Distributionen handelt, deren Vorteile aber in einem Bereich liegen, dessen Bedeutung für den universitären Bereich marginal ist.

Fedora, CentOS

fedoraproject.org, www.centos.org

Als nicht kommerzielle Schiene bzw. Ableger von Red Hat stellen diese Distributionen Entwicklungslinien dar, die viele Eigenschaften von Red Hat mit besserer Unterstützung neuer Hardware verbinden. Ansonsten gilt für diese beiden Distributionen das im vorigen Abschnitt Gesagte.

Debian GNU/Linux

www.debian.org

Die nicht-kommerzielle Debian GNU/Linux Distribution wird von einer demokratisch organisierten weltweiten Community entwickelt. Dabei wird nach dem Debian-Gesellschaftsvertrag vorgegangen, der unter anderem erfordert, dass Software, Dokumentation und auch alle anderen Komponenten der Distribution frei sein müssen. Das Debian-Projekt legt hier striktere Maßstäbe als so manch andere Distribution an. Bei Debian werden auch Sicherheitsprobleme öffentlich diskutiert, was im Allgemeinen zu einer raschen Behebung von Sicherheitslücken führt. Eine der Stärken von Debian ist die mächtige Paketverwaltung APT, mit der sehr bequem Software installiert werden kann und sich auch ein Upgrade der gesamten Distribution zu einer neueren Release einfach gestaltet. Debian GNU/Linux ist stets in drei Releases erhältlich:

- *stable*: (derzeit Debian GNU/Linux 3.1 'sarge')
  Die *stable* Distribution ist die aktuelle, offiziell freigegebene Distribution. Sie ist die Produktionsversion von Debian. Das gesamte System ist genauestens getestet und aufeinander abgestimmt. Nach einer Release von *stable* bleiben die Versionen der Softwarepakete konstant bis auf Sicherheits-Updates. Diese sind jedoch im Allgemeinen rasch verfügbar. *Stable* empfiehlt sich daher für den Einsatz auf Servern und anderen Produktionssystemen, die lange Zeit ohne größere Eingriffe laufen sollen. Da *stable* wegen der umfangreichen Testarbeiten aber nur relativ selten *released* wird, sind die Softwarepakete meist nicht sehr aktuell und der Einsatz auf sehr moderner Hardware könnte sich wegen mangelnder Hardwareunterstützung problematisch gestalten. Backports (z. B. www.backports.org) können in diesem Fall sehr hilfreich sein, wenn man trotzdem *stable* einsetzen möchte.

- *testing*: (derzeit Debian GNU/Linux 4.0 'etch', wird voraussichtlich noch Ende 2006 *stable* werden)
  Dies ist der Entwicklungszweig für die kommende *stable* Distribution. Pakete landen hier, nachdem sie eine gewisse Zeit in *unstable* getestet wurden und alle Abhängigkeiten erfüllen. Viele Debian-User setzen *testing* gerne auf ihren Arbeitsplätzen ein, wenn sie mehr Wert auf Aktualität (sowohl Softwarepakete, als auch Unterstützung neuerer Hardware) und weniger Wert auf Stabilität legen. Diese Vorgangsweise ist ganz besonders beliebt, wenn die aktuelle *stable* Release alt ist und eine neue Release bevorsteht.

- *unstable*: ('sid')
  Hier findet aktive Entwicklung statt. In diesem Zweig werden neue Versionen von Paketen und auch komplett neue Pakete zuerst aufgenommen. Hier werden sie auf Fehler geprüft. Es existieren in *unstable* zwar die aktuellsten Paketversionen, aber User müssen immer wieder mit Fehlern oder nicht erfüllten Paketabhängigkeiten rechnen. *Unstable* ist daher kaum für den produktiven Einsatz geeignet, sondern für das Testen von Paketen. *Unstable* wird daher von Debian-Entwicklern eingesetzt. Auch User, die zu Gunsten von Aktualität Stabilität opfern, sind mit Debian *unstable* gut bedient.

Ubuntu

www.ubuntu.com

Dieser in den letzten Jahren sehr populär gewordene Debian-Ableger erfreut sich immer größerer Beliebtheit und ist auch mit allen Nebenlinien, wie Kubuntu (www.kubuntu.org) und Xubuntu (www.xubuntu.org) an der TU weit verbreitet. Ubuntu und die Nebenlinien unterscheiden sich im Wesentlichen eigentlich nur durch die voreingestellte grafische Benutzeroberfläche (ubuntu/gnome, kubuntu/KDE bzw. xubuntu/Xfce). Die Hersteller legten nach eigenen Angaben großes Gewicht auf Benutzerfreundlichkeit und Sicherheit. Erwähnenswert ist die „Besonderheit", dass der Root-Account deaktiviert ist und administrative Aufgaben mit dem Befehl *sudo* durchgeführt werden.

Ansonsten gelten im Wesentlichen als Debian-Derivat die Aussagen über Debian.

Gentoo Linux

www.gentoo.org

Diese in so genannten Hardcore-Linux-Kreisen beliebte Metadistribution ist nur der Vollständigkeit halber angeführt. Es handelt sich dabei um eine in Quellform vorliegende Distribution, die der Benutzer selbst kompilieren muss. Detailliertes Wissen über die Hardware und deren Eigenschaften und tief gehendes Know-how über die inneren Strukturen von Linux sind dabei unbedingte Voraussetzung und daher stellt diese Linuxversion für die Zielgruppe dieses Artikels nur eine ziemlich theoretische Möglichkeit dar.

Slackware

www.slackware.com

Slackware ist eine Distribution, die sich durch Einfachheit und Stabilität auszeichnet. Sie kommt mit zwei CDs aus (eigentlich mit einer, wenn man keinen KDE braucht). Da die meisten Konfigurationsaufgaben mittels eines Texteditors zu bewerkstelligen sind, ist sie allerdings für den typischen Desktop-User weniger geeignet, umso mehr aber für kleinere Server-Anwendungen, ebenso für „embedded"-Anwendungen, wie Firewalls u.dgl. Es lassen sich auch grafische Oberflächen wie X11 mit KDE problemlos installieren, nur sollte man sich keine grafischen Systemadministrationstools erwarten. Das hat aber den Vorteil, dass es bei der Systemadministration keine Benutzerbevormundung gibt, die schon so manchem Systemadministrator (vor allem, wenn er gewohnt ist, mitzudenken) das Leben schwer gemacht hat. Aller-

dings muss man, wenn man die Systemdisk als RAID1 (gespiegelt) konfigurieren will, alles manuell machen.

Die Hardware-Erkennung („hotplug“) funktioniert sehr gut, und alles in allem ist Slackware eine altbewährte (eine der ersten Distributionen, die es gab), stabile und einfache Distribution, die für geübte Linux/Unix-Administratoren sehr zu empfehlen ist.

Zwar nicht Linux, aber Open-Source: BSD-Unix

www.bsd.org

BSD-Unixe gibt es (historisch gewachsen) in drei Varianten: FreeBSD, OpenBSD und NetBSD.

Um einen schnellen Eindruck über BSD zu gewinnen, habe wir uns FreeBSD 6.0 näher angesehen. Die Auswahl erfolgte auf Grund der ausgezeichneten Online-Dokumentation von FreeBSD, was eine kurze Einarbeitungszeit erwarten ließ.

Dazu wurde eine Standard-Server-Konfiguration testweise installiert, also sshd, Mailserver (sendmail, postfix, pop, imap), apache mit php und perl, die MySQL-Datenbank, ein Samba-Server und nfs.

Außerdem wurde X11 und KDE getestet, sowie der äußerst flexible und leistungsfähige Paketfilter „pf“.

Insgesamt konnte FreeBSD alle Erwartungen erfüllen, die Einarbeitungszeit war tatsächlich sehr kurz und alle Dienste liefen über lange Zeit einwandfrei und stabil. Die Hardware-Erkennung funktionierte überraschend gut, hier zeigte sich der Eindruck, dass FreeBSD doch sehr „aus einem Guss“ gefertigt ist. Selbst der Test auf einer „embedded hardware“, ein Kästchen mit Intel-kompatiblem Prozessor und CF-Disk, bei dem man das Betriebssystem in read-only-Bereiche und read-write-Bereiche aufteilen muss, verlief sehr zufrieden stellend, der Arbeitsaufwand hielt sich in Grenzen. Zusammen mit dem Paketfilter „pf“ macht dies das Betriebssystem sehr interessant für kleine Firewall-Anwendungen.

Auch X11 und KDE waren leicht zu installieren (auf einem Notebook, inklusive WLAN, getestet).

Für das Einspielen von Softwarepaketen gibt es zwei Möglichkeiten: Binärpakete, die auch remote eingespielt werden können (es werden ähnlich wie bei Debian Abhängigkeiten aufgelöst) sowie Sourcepakete, deren Makefiles auf /usr/ports liegen. Der Rest wird auch von einem Remote-Server kopiert, dann übersetzt und installiert (ähnlich Gentoo-Linux).

Fazit: Für erfahrene Linux/Unix-Administratoren ist FreeBSD eine wirklich ernst zu nehmende Alternative zu Linux.

Da das Systemmanagement in erster Linie (ähnlich wie bei Slackware-Linux, das auch wegen seiner Einfachheit und Stabilität besticht) mittels Texteditor und Konfigurationsdateien bewerkstelligt wird, ist es allerdings für eingefleischte Windows-User, die alles über grafische Management-Tools machen wollen, sicher kein Thema. Dafür gibt es nicht die leidige „Benutzerbevormundung“, die bei vielen kommerziellen Distributionen einem Systemadministrator das Leben schwer machen kann. Für Leute mit Linux- bzw. Unix-Erfahrung lässt sich FreeBSD allerdings auch als Desktop-Rechner zufrieden stellend einsetzen.

Nachteile sind: keine Journal-Filesysteme, UFS mit soft-Updates ist aber sehr stabil; für Linux-Binaries muss ein Emulator installiert werden; keine SYS-V Start-up-scripts (S99xxx, K99xxx u.dgl.), ähnlich wie bei Slackware (macht aber nicht wirklich Probleme).

Ein ähnlicher Test mit OpenBSD, welches besonders interessant für sicherheitsrelevante Anwendungen sein soll, ist noch in Planung.

Neben den beschriebenen Distributionen, die alle mehr oder minder als Universalbetriebssysteme sowohl für Server- als auch für Desktop-Anwendungen in Frage kommen und eine „komplette“ Palette von Softwareprodukten anbieten, sollen hier auch „Sonderdistributionen“ nicht unter den Tisch fallen. Unter anderen sind an dieser Stelle *damnsmalllinux* (www.damnsmallinux.org), *knoppix* (www.knoppix.org) und *grml* (www.grml.org) zu erwähnen. Diese auf Debian basierenden Linux-Versionen bieten in Spezialfällen (z. B.: eingeschränkte Hardware-Ressourcen) oder zur Behandlung von Hardwareproblemen (Rescue-Funktion) hochinteressante Möglichkeiten an.

Für alle angeführten Linux-Distributionen existieren Mirrors am Goodie Domain Service (GDS) der TU Wien (gd.tuwien.ac.at).

Abschließend ist es notwendig, mit Nachdruck darauf hinzuweisen, dass diese Zusammenstellung von Linux-Versionen nur einen Snapshot darstellt und nicht notwendigerweise eine lange Gültigkeit haben muss. Außerdem sind naturgemäß persönliche Erfahrungen eingeflossen und, wie bereits erwähnt, wird keinerlei Anspruch auf Vollständigkeit erhoben.

Skype pro & contra

Ein Beitrag zur Diskussion über kostenloses Telefonieren im Internet

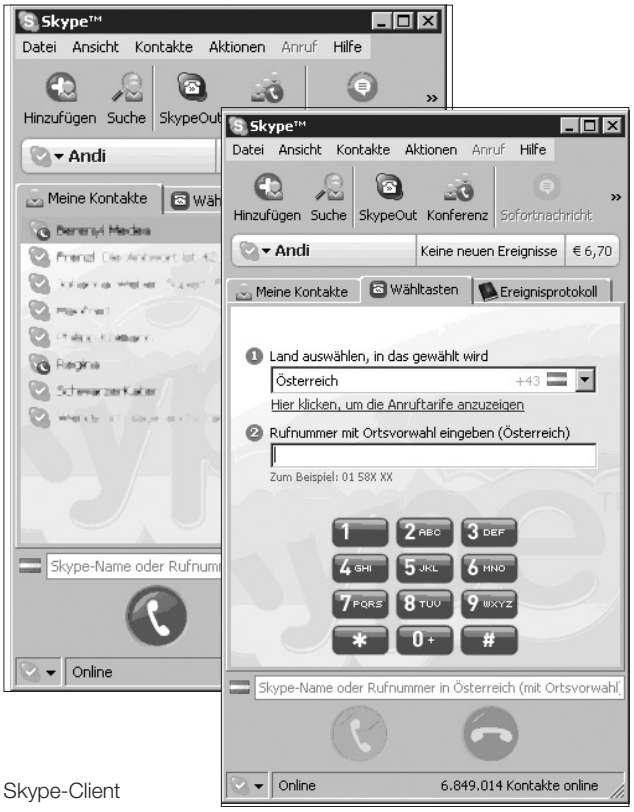
Andreas Klauda

Was ist Skype?

Skype – eine Software zum Telefonieren vom PC aus – ähnelt auf den ersten Blick sehr einem gewöhnlichen Instant Messenger wie etwa ICQ oder dem MSN-Messenger. Dieser Eindruck täuscht allerdings, da hierbei die grundlegende Funktionsweise, die Skype von allen anderen dieser Programme deutlich abhebt, missachtet wird:

Skype basiert auf Peer-to-Peer-Technologie, d. h. ähnlich wie Filesharing-Netzwerke verbindet sich jeder einzelne Client (also jedes einzelne Softphone) nur zu Authentifizierungszwecken mit einem zentralen Server, alle anderen Verbindungen werden mit anderen Clients („Peers“) direkt eingegangen, weswegen mit Skype diese sinnvollerweise auch verschlüsselt werden. Das Skype-Netz ist die Summe aller Skype-Clients. Anders als herkömmliche Telefonnetze wachsen Leistungsfähigkeit und Ausfallsicherheit des Skype-Netzes mit der Anzahl der Nutzer, da jeder neue Nutzer auch wieder seine eigenen Ressourcen einbringt (Rechenleistung, Bandbreite etc.), darum funktioniert es auch perfekt hinter Firewalls und NAT-Routern. Diese Eigenheit bringt nicht nur Vorteile sondern auch „Gefahren“ mit sich (mehr dazu im Abschnitt „Richtiger Umgang mit Skype“ und „Technische Details“).

Skype hat vor allem deswegen von sich reden gemacht, weil es mit Skype relativ häufig möglich ist, auch zwischen Clients, die durch eine Firewall bzw. ein NAT-System geschützt sind, VoIP-Verbindungen herzustellen, was bei SIP-basiertem[1] VoIP oftmals problematisch ist. Denn möchte Nutzer A (in Fachjargon: Alice) mit Nutzer B (im Fachjargon: Bob) sprechen und sind beide (!) durch eine Firewall/NAT geschützt, sind in keiner Richtung eingehende Verbindungen möglich. Ist dagegen nur Alice hinter einer Firewall, könnte in der Regel zumindest Alice bei Bob anrufen (allerdings wird bei Alice möglicherweise der Ton „geblockt“, d. h. der Ton funktioniert nur in eine Richtung), aber nicht Bob bei Alice.



Skype-Client

Jeder Skype-Client ist mit einem Supernode verbunden, der letztlich als Hub fungiert. Jeder Skype-Client kann ein Supernode sein. Supernodes sind also keine zentralen Server, sondern andere Skype-Nutzer.

Wenn ein Anruf getätigt wird, wird über die mit dem Supernode (C bzw. Charlie) dauerhaft bestehende TCP-

[1] Für die Internet-Telefonie wird in der Regel ein Protokoll namens SIP verwendet.
SIP steht für *Session Initiation Protocol*, was Sitzungs-Einleitungs-Protokoll bedeutet. Wie dieser Name schon richtig beschreibt, ist es nur zum Einleiten der Sitzung, also des Telefongesprächs verantwortlich, für die Übertragung der Daten an sich kommt ein weiteres Protokoll ins Spiel, welches SDP-Protokoll genannt wird. Diese Abkürzung steht für *Session Description Protocol*, die Session (wieder ist unser Telefongespräch gemeint) wird hiermit also beschrieben. Folglich werden mit diesem Protokoll die wichtigsten Daten des Gesprächs übermittelt, nicht aber die eigentlichen Audiosignale! Diese werden mit Hilfe des so genannten *Realtime Transport Protocol* (RTP) übertragen, welches die Daten in Pakete schnürt und per UDP versendet (UDP, da dieses Protokoll im Vergleich zu TCP/IP ein geringeres Datenvolumen-Aufkommen ermöglicht, was bei Telefongesprächen nicht unwichtig ist). Achtung: Skype verwendet nicht das SIP-Protokoll, es verwendet ein eigenes Protokoll.

Verbindung über das *Skype Proprietary Call Control Protocol* ein eingehender Anruf signalisiert. Anschließend verbinden sich Alice und Bob mit dem Supernode Charlie und können über Charlie eine Gesprächsverbindung aufbauen.

Supernodes kann man sich als Vermittlungsrechner vorstellen, die (in der Skype-Terminologie gesprochen) für die Global-Index-Funktion zuständig sind.

Mit Skype ist es möglich, mittels der Technik *Voice over IP* mit dem Gegenüber zu telefonieren. Dieses Telefonieren ist kostenlos, sofern es zwischen zwei Skype-Nutzern geführt wird.

Allerdings gibt es auch die Möglichkeit, von Skype auf ein gewöhnliches Telefon (Handy oder Festnetz) anzurufen.

Das Zauberwort hierfür lautet SkypeOut. Man kauft sich, wie bei einem Prepaid-Handy, ein bestimmtes Kontingent von der Skype Homepage, und kann nun den Computer als gewöhnliches Telefon nutzen.

Das Gleiche funktioniert auch umgekehrt. Bei SkypeIn erhält man eine in einem bestimmten Land lokalisierte Festnetz-Nummer, welche von einem beliebigen Telefon aus angerufen werden kann, der Anruf erscheint dann im Skype-Fenster.

Eine weitere Funktion des Programms ist es, Anrufe in Abwesenheit wie bei einem gewöhnlichen Telefon mittels Anrufbeantworter annehmen zu lassen. SkypeIn sowie die Anrufbeantworter-Funktion sind allerdings kostenpflichtig. (Es wird eine monatliche Grundgebühr eingehoben. Pro Gespräch fallen weitere Kosten an.)

Neben diesen Funktionen ist es mit Skype auch möglich, Audiokonferenzen zu führen, maximal 10 Personen können daran teilnehmen.

Entwickelt wurde Skype von den KaZaa-Entwicklern Niklas Zennström und Janus Friis, das ist wahrscheinlich auch der Grund für die Peer-to-Peer Technologie.

Was ist SkypeOut?

Mit der Basis-Version von Skype ist es nur möglich, mit anderen Skypern zu chatten oder sie anzurufen. Beides ist, abgesehen von den Strom- und Internet-Kosten, vollkommen gratis. Eine sinnvolle Erweiterung des Prinzips ist hier nun SkypeOut.

SkypeOut ermöglicht Anrufe ins Festnetz sowie ins Mobilfunknetz, weltweit. Ausgenommen sind lediglich Service-Rufnummern, natürlich insbesondere solche, die sonst die Telefonrechnung zusätzlich belasten würden (Mehrwert-Rufnummern).

Soweit besteht noch kein Grund, der einen davon abbringen könnte, das handelsübliche Telefon für seine Gespräche zu verwenden; Das Besondere bei SkypeOut sind jedoch die Kosten: Für Gespräche in die gängigsten Ziele werden nur 2 Cent pro Minute berechnet (inklusive 15% belgischer Mehrwertsteuer, da Skype in Belgien bilanziert).

Folgende Länder haben diesen günstigen Tarif: Argentinien (Buenos Aires), Australien, Österreich, Belgien, Kanada, Kanada (Mobiltelefone), Chile, China (Beijing, Guanzhou, Shanghai, Shenzhen), China (Mobiltelefone), Dänemark, Frankreich, Deutschland, Griechenland, Hong Kong, Hong Kong (Mobiltelefone), Irland, Italien, Mexiko (Mexiko City, Monterrey), Die Niederlande, Neuseeland, Norwegen, Polen (Gdansk, Warschau), Portugal, Russland (Moskau, St. Petersburg), Spanien, Schweden, die Schweiz, Taiwan (Taipei), England, die USA (außer Alaska und Hawaii), die USA (Handys) und der Vatikan.

Die Tarife für alle weiteren Länder kann man hier einsehen: www.skype.com/products/skypeout/rates/all_rates.html.

Bei SkypeOut muss man sein Guthaben, wie bei gängigen Prepaid-Handys, im Voraus aufladen. Es gibt 10-Euro-Pakete im Angebot.

Bezahlen kann man per Paypal, Visa, Eurocard/Mastercard, Online-Bankbuchung, normaler Banküberweisung, Dinersclub Karte oder Moneybookers.

Was ist SkypeIn?

Mit SkypeOut ist es möglich, von Skype aus ins Festnetz anzurufen, entsprechend kann man mit SkypeIn vom Festnetz auf einen Skype-Account anrufen.

Das funktioniert so: man logt sich auf der Homepage der Skype-Entwickler ein (www.skype.com) und bestellt sich dort eine Rufnummer. Es stehen verschiedene Länder zur Verfügung, wobei der Bestellvorgang für jedes Land ein anderer ist (manche Länder darf man beispielsweise nur als Bürger des entsprechenden Landes auswählen). Dies kann den Vorteil haben, dass Sie z. B. für Ihre amerikanischen Freunde unter einer amerikanischen Festnetznummer erreichbar sind, aber hier in Österreich sitzen. Für den Gesprächspartner ist es eventuell nur ein lokales Gespräch und daher um vieles günstiger als direkt in Österreich anzurufen. Pro Skype-Account sind bis zu 10 SkypeIn-Telefonnummern registrierbar.

Der Preis für eine SkypeIn Nummer beträgt 10 Euro für 3 Monate, oder 30 Euro für 12 Monate.

Im Besitz einer solchen Rufnummer kann man angerufen werden, egal wo man gerade ist. Wo man die Skype-In-Nummer bestellt hat, spielt an sich keine Rolle, nur dass das Telefongespräch des Anrufers so abgerechnet wird, als würde er eine Nummer in dem entsprechenden Ort anrufen.

Richtiger Umgang mit Skype

Supernodes sind für das TUNET natürlich problematisch, wegen der hohen Anzahl an Verbindungen und dem Traffic, der entsteht.

Es gibt jedoch einfache Verhaltensregeln im Umgang mit Skype, um dies zu verhindern:

- Skype **nicht** über Nacht laufen lassen, sondern beenden, wenn der Client nicht mehr benötigt wird.
- Skype **nie** auf Rechnern (Server) installieren, auf denen kein Benutzer lokal arbeitet.

Um als Supernode zu fungieren (besser gesagt: fungieren zu dürfen!), benötigt man eine (sehr) gute Inter-

net-Anbindung (256 kbit/s oder mehr im Upload), der Computer muss eingehende Verbindungen akzeptieren (Rechner hinter NAT-Routern werden somit niemals Supernodes) und Skype muss tage- bzw. wochenlang ohne Neustart laufen. Wenn man also die beiden obigen Verhaltensregeln beachtet, wird der Skype-Client **kein** Supernode.

Um mit Skype komfortabel zu telefonieren ist ein Headset sehr zu empfehlen, über Mikrofon und Lautsprecher funktioniert es zur Not auch, aber dann kann es zu störenden Echos bzw. Rückkoppelungen kommen.

Technische Details

Steuerung von NAT/Firewall durch Skype

„Über Supernode eine Gesprächsverbindung aufbauen" ist eine freundliche Umschreibung für den kreativen Umgang mit den beteiligten NAT- und Firewall-Systemen. Der Supernode kennt auf Grund der zu ihm von Nutzer A und Nutzer B bestehenden TCP-Verbindung die Ports, die Nutzer A und Nutzer B jeweils zum Versenden der Audiodaten verwenden.

Die folgende Technik wird *UDP hole punching*[2] genannt und Skype bedient sich ihrer, um trotz Firewall auf beiden Seiten die Daten direkt von A nach B zu bringen. Der von Nutzer A angerufene Nutzer B schickt zunächst ein UDP-Paket an den ihm durch den Supernode mitgeteilten Port, den Nutzer A für ausgehende Verbindungen nutzt. Dieses Datenpaket wird auf der Firewall von Nutzer A blockiert, da es sich um ein unaufgefordert von außen gesendetes Datenpaket handelt. Die Firewall bei Nutzer B merkt sich aber, dass über diesen Port ein abgehendes Paket an Nutzer A gesendet wurde und öffnet daher diesen Port für ankommende Datenpakete von Nutzer A. Dieser sendet nun auf diesen bei Nutzer B geöffneten Port ein Paket, dort wird es als gültige Antwort auf die zuerst geöffnete Verbindung gewertet und somit sind auf beiden Firewalls die Ports geöffnet.

Audio-Übertragung

Dieser Mechanismus dient nur zum Aufbauen der Verbindung. Das eigentliche Gespräch läuft dann direkt zwischen Nutzer A und Nutzer B (normalerweise per UDP). Diese Vorgehensweise hat zumindest gewisse Ähnlichkeiten mit einem STUN-Protokoll[3]. Teilweise wird daher vermutet, Skype nutze (u.a.) ein modifiziertes STUN-Protokoll. Allgemein gesprochen sind die in Skype eingesetzten Methoden wie auch STUN-Technik dem *UDP hole punching* zuzuordnen.

Relaying

Führt die eben beschriebene Methode bei besonders restriktiven Netzwerkkonstellationen nicht zum Erfolg, geht Skype noch einen Schritt weiter: Das Gespräch wird dann tatsächlich über andere Computer (D) geleitet (*relaying*). Idealerweise sollte die eingesetzte Firewall zumindest ausgehenden UDP-Traffic zulassen – was häufig der Fall sein wird. D wird vom Supernode in seiner Funktion als Verwalter des bereits erwähnten Global Index ausgewählt.

Aber selbst wenn in Ausnahmefällen ausgehender UDP-Traffic von der Firewall blockiert wird, kann Skype trotzdem häufig noch funktionieren, indem Skype komplett auf TCP umschwenkt. Ausweislich des auf www.skype.com als PDF veröffentlichten „Guide for Network Administrators" ist Skype nicht auf UDP angewiesen: „*Skype will work fine without the ability to transmit UDP messages".* Die Tonqualität wird freilich von reduzierter Qualität sein.

Relaying wird aber nur über Computer stattfinden, die über eine relativ große Upload-Bandbreite verfügen und bei denen keine Firewall eingehende TCP/UDP-Verbindungen auf dem in Skype festgelegten Port unterbindet.

Das Relaying ist übrigens nicht nur für die Audio-Übertragung, sondern auch für eine eventuelle Dateiübertragung (*file transfer*) nützlich. Zwar ist die Übertragungsrate dann sehr niedrig, aber immerhin kommt die Dateiübertragung (anders als mit manch anderen Programmen) überhaupt zustande.

Spiegel-Interview mit dem Skype Manager

vom 30.8.2006
Komplettes Interview unter http://www.spiegel.de/netzwelt/technologie/0,1518,434092,00.html

SPIEGEL ONLINE: *Administratoren in Unternehmen mögen Skype überhaupt nicht. Sie sagen, die Software durchlöchere die Firewall, wenn man sie installiert.*

Jackson: *Zuallererst: Skype reißt keine Löcher in Firewalls. Es öffnet keine Ports, es nutzt vielmehr Ports, die schon offen sind. Warum machen wir das? Wir wollten, dass die Software leicht zu installieren ist und auf Anhieb läuft. Und so sucht sich die Software nach der Installation selbst den Weg ins Internet. Wenn ein Administrator also eine Lücke in seiner Firewall gelassen hat, dann wird Skype diese nutzen.*

SPIEGEL ONLINE: *Aber daran stören sich Administratoren, weil sie keinen Einfluss darauf haben.*

Jackson: *Richtig. Wir erweitern die Software gerade um Funktionen, mit denen Administratoren Skype besser kontrollieren können. Immerhin wird Skype ja inzwischen zu 30 Prozent geschäftlich genutzt. Die Administratoren können dann den Port einstellen, den das Programm nutzen soll. Andere Ports, die vorher schon offen waren, werden so natürlich nicht geschlossen.*

[2] NAT Traversal über STUN wird auch als UDP hole punching bezeichnet und funktioniert bei den folgenden drei NAT Typen: Full Cone NAT, Restricted Cone NAT und Port-Restricted Cone NAT. Andere NAT Umgebungen und auch NAT Traversal zwischen symmetrischen NAT werden nicht unterstützt.

[3] STUN (Simple Traversal of UDP Through NATs) - Über das STUN-Protokoll bzw. durch einen STUN-Server können Probleme, die sich durch Router und Firewalls ergeben, umgangen werden. Ein Router oder eine Firewall verbergen ein internes Netzwerk vor dem Internet, in dem sie interne Netzwerkadressen nach außen umwandeln (NAT = Network Adress Translation). Ein wichtiges Ziel beim STUN ist es zunächst einmal, überhaupt mitzukriegen, welche Art von Firewall vorliegt, dann kann eine Ausweichstrategie gefunden werden. Hierbei wird eine Anfrage an einen öffentlichen STUN-Server gestellt („wie lautet meine öffentliche IP-Adresse und Port"), diese Information kann dann in die VoIP-Header anstatt der privaten Adressen eingebaut werden.

CMS für die TU-Website

Werner F. Sommer, PR und Kommunikation

In meinem letzten Artikel zur TU-Website (ZIDline 5, Juni 2001) hatte ich geschrieben: „Es stünde der *Technischen* Universität gut zu Gesicht, eine Website auf der Höhe der Zeit zu haben." Gut Ding braucht offensichtlich Weile, vor allem an einer Universität, selbst wenn es eine technische ist.

Schwer abzuschätzen, wie viel Millionen Zeichen HTML im letzten Jahrzehnt für die TU-Website manuell geschrieben oder via Copy & Paste vervielfältigt wurden. Eine Arbeitsweise, die einerseits spezifische Kenntnisse erfordert, andererseits mühselig, ineffizient und fehleranfällig ist.

Zeitgemäßes Publizieren via CMS

Ergo lag es nah, sich um eine zeitgemäße Lösung dieses Problems zu kümmern. Das Kürzel dafür lautet CMS, Content Management System. Damit wird es möglich, ohne HTML-Kenntnisse Inhalte schnell und konsistent im Web zu publizieren. Die konkreten Vorteile des Publizierens von Webinhalten via CMS gegenüber der „Do it yourself"-Methode aus meiner Sicht:

- MitarbeiterInnen ohne HTML-Kenntnisse können Inhalte einpflegen, d. h. die Information kann dort erfasst werden, wo auch der entsprechende Sachverstand sitzt und muss nicht den Umweg über HTML-kundige SpezialistInnen gehen. Mittelfristig wird das den Inhalt der Site qualitativ und quantitativ verbessern.
- Die Rollenverteilung innerhalb des Systems (AdministratorInnen, RedakteurInnen usw.) kann präzise definiert werden. So werden alle nur damit konfrontiert, womit sie umgehen wollen und können. Im Zuge dieser Rollenaufteilung können auch komplette Workflows (z. B. Erfassung eines Texts, Korrekturschleifen, Freigabe) abgebildet werden.
- Ein einheitliches Look & Feel wird gewährleistet, weil die „RedakteurInnen" nur mit den zentral definierten Templates (Vorlagen) und Styles (Formatierungen) arbeiten können.
- Entsprechend kann so auch die „Barrierefreiheit", also die Zugänglichkeit der Inhalte für Menschen mit Behinderungen und chronischen Erkrankungen, besser gewährleistet werden.
- Inhalte können in verschiedenster Art und Weise ausgegeben werden (z. B. auch als WML für tragbare Geräte).
- Die Konsistenz der Inhalte steigt. Wird z. B. eine innerhalb des Systems referenzierte Seite gelöscht, verschwindet auch der Link darauf.
- Die Redundanz der Inhalte nimmt ab, weil Inhalte nur einmal eingepflegt werden müssen und dann an beliebig vielen Orten eingebunden werden können.
- Inhalte können durch Angabe von Start- und Ende-Datum zeitgesteuert publiziert werden. Die Archivierung erfolgt „automatisch".
- Das Tracking des UserInnenverhaltens (wer ruft wann welche Inhalte auf?) ist mit CMS weit komfortabler als ohne.
- Diverse „Plug Ins" erhöhen die Funktionalität der Site für die UserInnen.

Open Source CMS TYPO3

Hinsichtlich der Kosten war es möglich, im Rahmen des Programms „Finanzierungsanreize zur Förderung der Profilentwicklung der Universitäten" die erforderlichen Mittel unterzubringen. Hinsichtlich des Systems war klar, dass nur ein Open Source Produkt auch die technische Betreuung durch den Zentralen Informatikdienst (ZID) gewährleisten würde. Nach Durcharbeiten der Möglichkeiten im vergangenen Herbst fiel die Wahl ziemlich schnell auf TYPO3. Hier ist weltweit eine ausreichende Community am Werk (http://typo3.org/) und es gibt weltweit über 200.000 TYPO3-Installationen.

Schritt zwei war die Beschaffung entsprechender Expertise von außen. Hier diente die Implementierung an der Universität für Bodenkultur (Boku) als Beispiel: Das für die Boku tätige Unternehmen – plan2net (http://www.plan2net.at/) – definierte sich als „Coach". Dieses Rollenverständnis garantiert einerseits den Know-how-Transfer von den Profis zur TU-IT, andererseits können – durch die Einbeziehung der TU-Arbeitskraft – auch die Kosten niedrig gehalten werden.

Teamwork

Der Arbeitsauftrag an plan2net und das TU-interne Projektteam lautet, die bestehende Website – mit lediglich geringen Einschränkungen bzw. Modifikationen – 1:1 mit dem CMS zu hinterlegen. TU-seitig besteht das Team aus Bettina Neunteufl (Projektleiterin; Redaktion;

PR und Kommunikation), Alexander Rajkovats (Software; ZID), Irmgard Husinsky (Styles; ZID) und Michael Roth (Systemadministration; ZID). An dieser Stelle ein Dank an alle Beteiligten (auch über die Genannten hinaus) für ihren Einsatz und die jeweiligen Vorgesetzten für deren Verständnis.

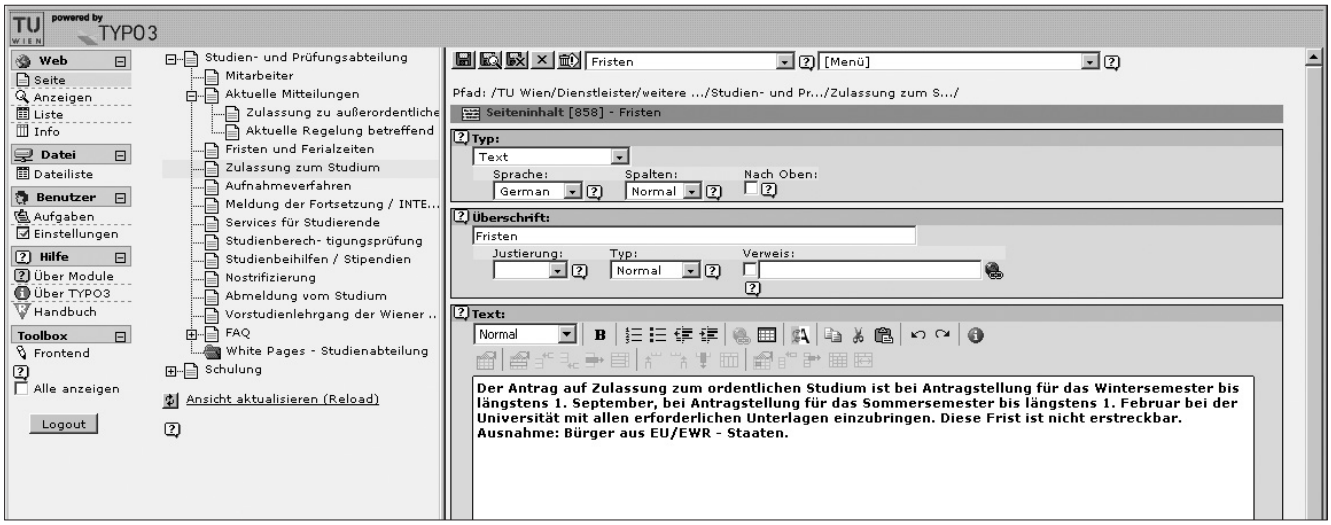
Der Zugang zum CMS erfolgt über TUWIS++, d. h. jene MitarbeiterInnen, die via TYPO3 Inhalte publizieren, haben auf der persönlichen TUWIS++-Startseite einen Link „CMS TYPO3“. Damit gelangen sie in das Backend der Applikation.

TYPO3 ist ein mächtiges CMS, dementsprechend ist auch die Bedienung nicht immer intuitiv zugänglich. Am 10. November wurden KollegInnen aus dem Bereich der ehemaligen Zentralen Verwaltung und des ehemaligen Außeninstituts mit der grundlegenden Bedienung vertraut gemacht. Binnen vier Wochen soll die Portation der Inhalte aus den aktuellen, statischen HTML-Seiten ins CMS erfolgen.

Das Projekt sollte eigentlich schon abgeschlossen sein. Allerdings haben sich die inhaltlichen (strukturellen) Probleme der Website einerseits und die personellen Umstellungen in PR und Kommunikation andererseits negativ auf den Zeitplan ausgewirkt. Das ehrgeizige Ziel, noch im laufenden Jahr mit der CMS-gestützten Site online zu gehen, bleibt aber aufrecht.

So alles gut läuft, werden also vor Jahresende noch wesentliche Teile der TU-Website über TYPO3 publiziert. Damit wäre die Site technisch wieder auf der Höhe der Zeit angelangt. Fast keine Universitätswebsite wird heutzutage noch ohne CMS betrieben. Freilich ist ein CMS aber nur ein technisches Hilfsmittel. Die Qualität und Quantität sowie der Nutzwert der Inhalte liegen in der Hand jener, die diese publizieren.

PS: Sobald das System stabil läuft, sind natürlich auch andere Organisationseinheiten eingeladen, ihre Webseiten via CMS zu betreiben!



Das TYPO3 Backend

Finite Elemente und Strömungsdynamik

Eine Erfolgsgeschichte

Peter Berger

Der Einsatz von (meist) kommerziellen Softwareprodukten aus den Bereichen Finite Elemente und Strömungsdynamik zieht sich wie ein roter Faden durch die Geschichte der verschiedenen Computersysteme an der TU Wien. Der Kauf des neuen IBM-Clustersystems (icp5.zserv) stellt bereits die 10. Generation von Computersystemen dar, auf denen diese Softwarepakete zum Einsatz kommen, und zeigt damit den hohen Stellenwert dieser Anwendungen für Forschung und Lehre.

Wie alles begann

Im Jahre 1978 wurden auf dem System CDC CYBER 74 die Programmpakete SAP 4 und NONSAP installiert (UC Berkeley, Berechnung von Tragwerken mit Hilfe der FE-Methode) und damit der erste Schritt zur Nutzung dieser Verfahren gesetzt. Diese Pakete (erweitert durch das Programmpaket STRESS der Uni Laibach) waren bis zum Abbau der CDC CYBER-Anlagen im Jahre 1992 in Verwendung. Das FE-Paket ADINA wurde 1988 auf der CDC CYBER 180-860 lizenziert.

Im Jahre 1988 wurden am Interuniversitären EDV-Zentrum (IEZ) an der TU Wien auf einem System NAS AS/9160 (ein Mainframesystem von Hitachi mit einer Vektorunit, 16 MB Hauptspeicher und einer floating-point Leistung von 17 MFlop/s) die FE-Pakete ABAQUS und ADINA installiert und für Festigkeitsberechnungen eingesetzt.

Die Installation des ersten Vektorrechners im Jahre 1990 (Siemens/Fujitsu VP50-EX mit 128 MB Hauptspeicher und einer floating-point Leistung von 238 MFlop/s) ermöglichte eine deutliche Ausweitung des Softwareangebots auf drei FE-Pakete (ABAQUS, ADINA und NISA II) und ein Softwarepaket für Strömungsdynamik (FIDAP).

Nach der Auflösung des IEZ und der Gründung des neuen EDV-Zentrums der TU Wien im Jahre 1991 endete auch die Ära der Mainframes, das Fachbereichsrechner-Konzept führte zur Installation der Systeme Convex C3220 und des „Fachbereichrechners Maschinenbau“ (IBM RS 6000-950 und 550). Dadurch konnten zu den bestehenden weitere FECFD-Pakete wie ANSYS, EMAS, NASTRAN, MARC lizenziert werden.

Die Ausweitung des Kooperationsvertrags zwischen der TU Wien und der Firma Siemens ermöglichte im Jahre 1992 den Austausch der VP50 auf einen Vektorrechner Siemens/Fujitsu S100 mit der ca. 3,5-fachen Leistung. Auf diesem System kam das Strömungspaket FIRE zum Einsatz, das vor allem in der Automobilindustrie (Motorenbau) verwendet wurde.

Der erste Applikationsserver FECFD

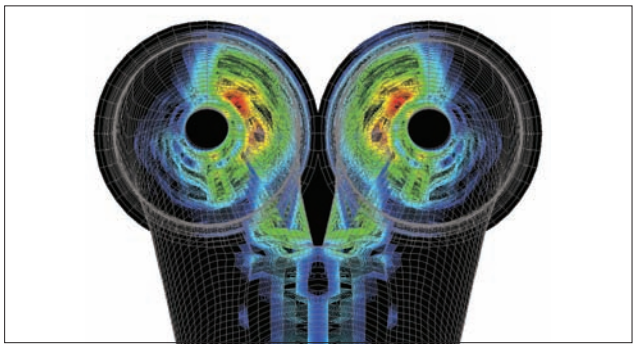
Im Jahre 1995 ging die Ära der Fachbereichsrechner zu Ende und wurde durch das „Applikationsserver-Konzept“ abgelöst. Kernaussagen dieses EDV-Konzepts, das in den wesentlichen Zügen auch heute noch gültig ist (und sich bestens bewährt hat), sind:

- für bestimmte Applikationen optimale Hardware- und Systemarchitekturen zur Verfügung zu stellen,
- deutlich höhere Performance (Memory und Massenspeicher) als Institutssysteme,
- kostengünstige Lizenzierung teurer kommerzieller Softwarepakete.

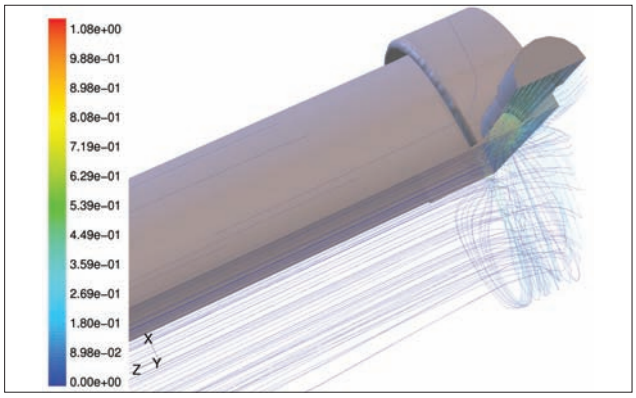
Einer der ersten Applikationsserver, der entsprechend diesem Konzept angekauft wurde, war 1995 der „Applikationsserver für Finite Elemente und Strömungsdynamik“. Dieses System der Firma Digital (DEC 8200 mit 2 CPUs und 2 GB Memory) wurde schrittweise ausgebaut und erweitert, sodass im Jahre 1998 der FECFD-Cluster bestehend aus zwei Maschinen (DEC 8200 und DEC 8400 mit insgesamt 12 CPUs und 16 GB Hauptspeicher) und den Software-Paketen ABAQUS, ANSYS, MARC, EMAS, FLUENT, FIDAP und CFX zur Verfügung stand.

Im Jahre 2002 wurde als Nachfolgesystem ein Clustersystem der Firma HP/Compaq gekauft (HP SC45-Cluster), das über 10 Knoten (ES 45 mit je 4 CPUs und

16 GB Memory), ein gemeinsames Disk-Storage (2,6 TB) und eine schnelle Kopplung (Quadrics, 360 MByte/s) verfügte. Dieses Clustersystem mit 40 CPUs unter dem Betriebssystem TRU64 UNIX verfügt auch bis zum heutigen Tag über gute Leistungswerte sowohl in Bezug auf die Rechenleistung (DEC-alpha CPUs mit 1 GHz) als auch auf die Clustersoftware (gemeinsames Filesystem, Batch- und Queue-Management, schnelle Kopplung, gutes Clustermanagement) und erfüllt heute vor allem im Bereich der Lehre wertvolle Dienste.

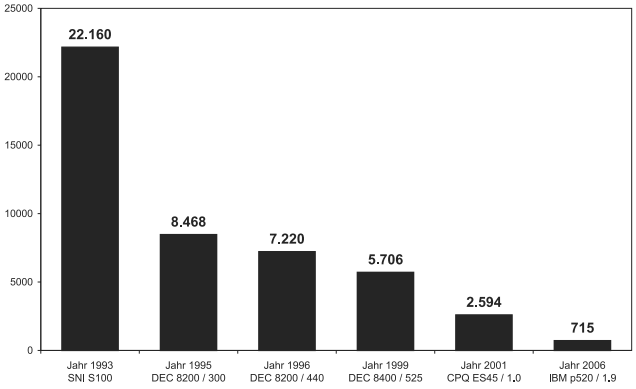


Wandfilmausdehnung im Einlasstrakt (Anwendung FIRE)
Aus: M. Klepatsch: Simulation der Mehrphasenströmung im Motor, Projekte an den zentralen Applikationsservern, Berichte 1997, EDV-Zentrum, TU Wien



Strömungslinien im Sekundärkreis eines Hämodialyse-Moduls (berechnet mit FLUENT 5.1).
Aus: M. Harasek, Das CDF-Paket FLUENT, ZIDline 2, Dezember 1999.

Das Diagramm zeigt die Entwicklung der CPU-Leistung der Systeme vom Jahr 1993 bis 2006 am Beispiel des ABAQUS-Standardbenchmarks STD-T4 (gemessene Laufzeiten auf den installierten Systemen in Sekunden).



Laufzeiten für den ABAQUS STD-T4 Benchmark (single CPU)

Der neue ZID-Cluster 2006 – FECFD

Im Frühjahr des vergangenen Jahres wurde mit der Planung einer Ersatzanschaffung für den über 3 Jahre alten Applikationsserver für Finite Elemente und Strömungsdynamik (SC45-Cluster) begonnen. Ausschlaggebend für diese Neuplanung war, dass für einen Großteil der installierten kommerziellen FECFD-Softwarepakete in naher Zukunft keine Updates und neue Versionen für das Betriebssystem TRU64 UNIX zur Verfügung stehen werden.

Im Jänner 2006 wurde eine EU-weite öffentliche Ausschreibung durchgeführt (maximaler Finanzrahmen € 380.000,- für die Rechnerhardware, für das Gesamtprojekt standen in Summe € 450.000,- zur Verfügung).

Die Veröffentlichung dieser Ausschreibung erfolgte am 13. Jänner 2006. Nach einer intensiven Prüfung der Angebote wurde am 5. April 2006 der Firma EDV-Design Informationstechnologie Ges.m.b.H. der Zuschlag für ein Clustersystem von IBM (POWER5+ Prozessoren) erteilt.

Die Cluster- und Hardwarearchitektur

Die Clusterknoten (Zugangsknoten und Compute-Nodes, Fileserver, Backup-Library und alle Netzwerkkomponenten) sind in 6 Schränken (19 Zoll) installiert. Alle 54 Compute-Nodes sind 4U-hohe Systeme mit 2 Prozessoren IBM POWER5+ (Dual-Core Module) mit 1,9 GHz Taktrate und 16 GB Hauptspeicher, der Zugangsknoten (fat node) verfügt über 8 CPUs IBM POWER5+ (2x Quad-Core Module) mit 1,5 GHz und 32 GB Memory. Das 2-Core p-520 System Planar enthält ein Dual-Core Module (DCM) und das lokale Memory-Subsystem.

Systemname: **icp5.zserv.tuwien.ac.at**

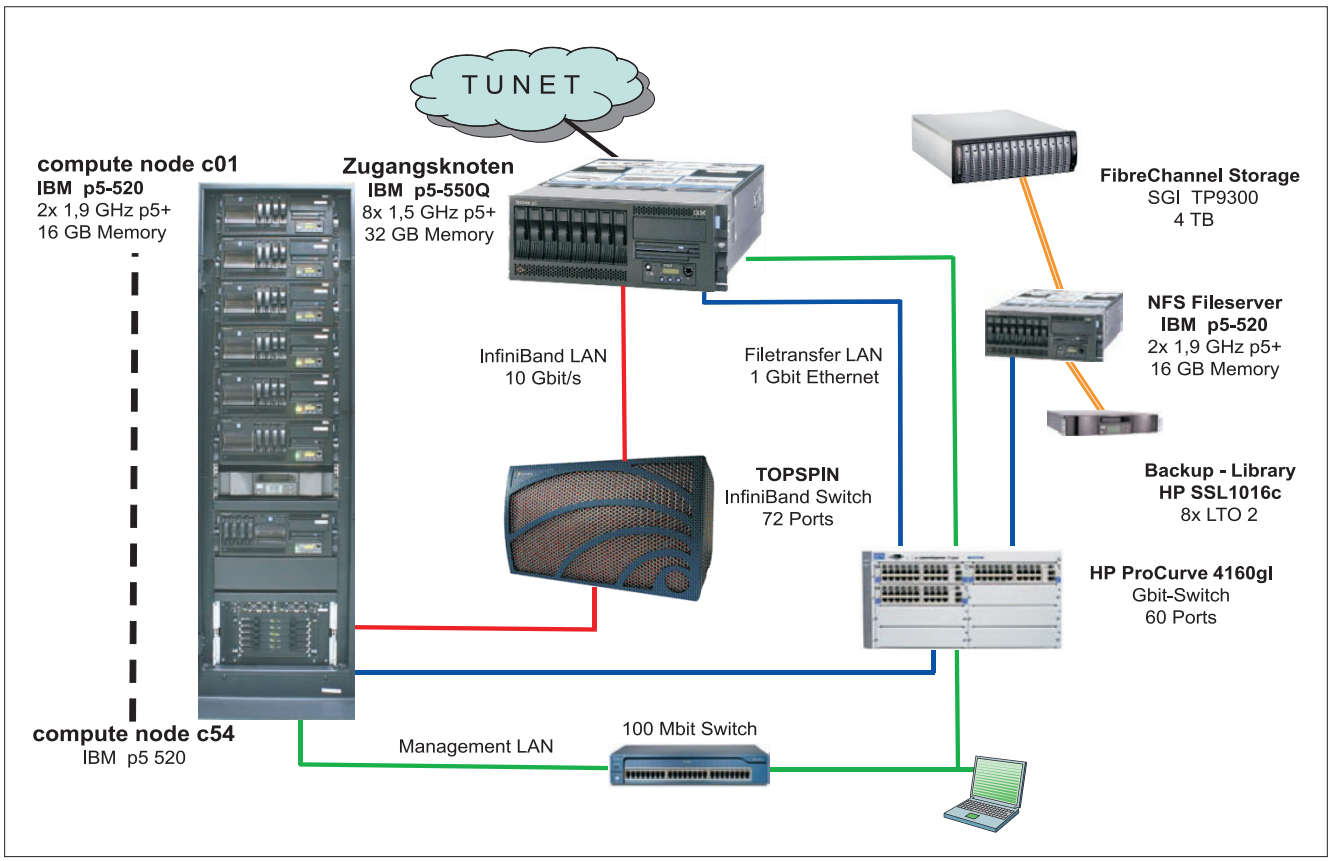
Das DCM enthält die beiden Prozessorkerne (1,9 GHz), den gemeinsamen L2 Cache (1,9 MB) und die Verbindung zum L3 Cache (36 MB, 30,4 GB/s Bandbreite). Der Chip ist in 90 nm CMOS-Technologie gefertigt und enthält Schnittstellen-Bausteine wie einen Memory-Controller, einen horizontalen und vertikalen Fabric Bus für den SMP-Support sowie eine GX+ Schnittstelle für I/O-Devices.

Das Memory-Subsystem besteht aus 8 Memory-Bänken (DDR2), die über 2 SMI-II Controller mit einer Bandbreite von 21 GB/s mit dem DCM verbunden sind.

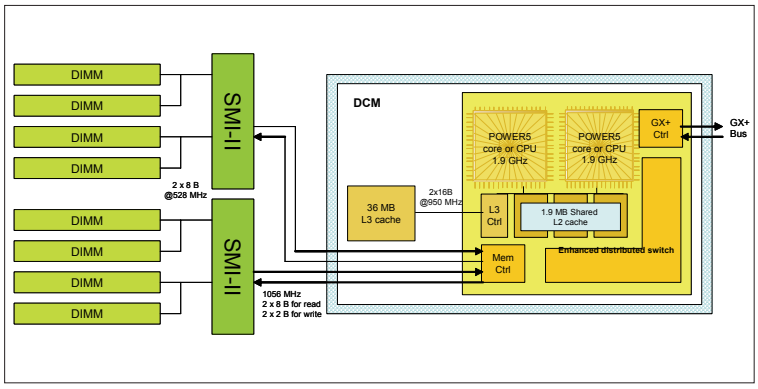
Die I/O-Schnittstelle ist der GX+ Controller, der über einen Enhanced I/O-Controller einen GX+ Port und 4x PCI-X Buses mit einer Bandbreite von 5 GB/s ansteuert. An diesem GX+ Port ist der InfiniBand Host Bus Adapter (HBA) angeschlossen.

Für die schnelle Kopplung der Clusterknoten mit MPI wird 4x InfiniBand (10 Gbit/s full duplex) von der Firma TOPSPIN (Cisco) eingesetzt. Für den interaktiven Zugang und den Filetransfer steht ein Gbit-LAN zur Verfügung.

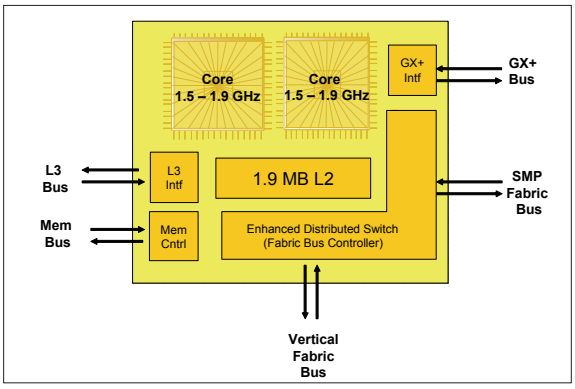
Drei Lizenzserver (ausfallssicher in drei Gebäuden aufgebaut) verteilen die erforderlichen Applikations-Software-Lizenzen auf die Clusterknoten.



Cluster-Konfiguration



Das p5-520 POWER5+ DCM



POWER5+ Chip

Installation, Abnahme und Benutzer-Testbetrieb

Der geplante Installationstermin im Mai 2006 konnte leider nicht eingehalten werden, da knapp vor der Auslieferung der Systeme eine technische Änderung der Konfiguration von IBM durchgeführt wurde (InfiniBand-Karte nicht am PCI-X sondern am GX+ Bus). Dadurch und durch eine sehr späte Lieferung des InfiniBand-Switches wurde die Installation erst Ende August 2006 abgeschlossen.

Nach der erfolgreichen Abnahme (Leistungstest und 4-wöchiger Dauertest) wurde der Testbetrieb gestartet, die erforderlichen Anpassungen der Software-Pakete durchgeführt und das Queueing-System an die Betriebserfordernisse angepasst.

Leider traten nach einigen Wochen Software-Probleme im Bereich InfiniBand auf, die nur durch die Einschaltung des IBM-Labors in Austin gelöst werden konnten. Das System läuft seit einigen Wochen (unter Hochlast) ohne Probleme.

Der Benutzer-Testbetrieb wird immer mehr ausgeweitet. Wir gehen davon aus, dass zum Jahreswechsel die Migration aller Benutzer des SC-Clusters auf das neue System abgeschlossen sein wird.

Die **Systembetreuung** wird von Josef Beiglböck (42071, beiglboeck@zid.tuwien.ac.at) und Dietmar Sonnleitner (42087, sonnleitner@zid.tuwien.ac.at) durchgeführt.

Weitere Informationen im **Web**:
http://www.zserv.tuwien.ac.at/icp5/

GemStone/S

Eine objektorientierte Datenbank

Thomas Mikulka

2002 kam es an der TU Wien zu einer grundlegenden Umstellung des „White Pages Service“ [1]. Neu am geänderten Konzept war u.a. die Verwendung einer ausgelagerten ZID Personendatenbank, die die Quelle aller „White Pages“-Einträge sein sollte. Schon wieder ein neuer Datenbank-Server? Nein, im Gegenteil! Hinter dem Begriff „Personendatenbank“ steckt tatsächlich eine weitere Teilfunktion einer erfolgreichen und höchst ungewöhnlichen Datenbank. Ein Überblick.

Basistechnologien und Funktionsprinzip

Die ZID Personendatenbank [2] ist eine Teilfunktion einer Datenbank, die sich bereits seit 1993 an der TU Wien im Einsatz befindet. Sie ist, im Gegensatz zu den heute gängigen, relationalen Systemen (z. B. Oracle, MySQL oder PostgreSQL etc.), eine rein objektorientierte Datenbank. Hersteller ist die Firma GemStone Systems [3] mit Sitz in Beaverton, Oregon, der offizielle Produktname lautet „GemStone/S Object Server“. Das beigefügte „S“ nach „GemStone“ deutet bereits auf eine ihrer markantesten Besonderheiten hin: die Datenbank selbst, sowie alle TU-spezifischen Applikationen, sind – bis auf wenige Ausnahmen – in Smalltalk geschrieben worden. Was sind nun die Basistechnologien dieser Datenbank?

Objektorientierung

Der wesentliche Unterschied zwischen einer GemStone/S und einer relationalen Datenbank ist: in GemStone/S ist jeder Dateneintrag ein Objekt und nicht zwangsläufig ein Tabelleneintrag, daher der Name „Object Server“. Mehr noch: alles, was sich begrifflich ansprechen lässt, ist in Smalltalk/GemStone ein Objekt. Ein Objekt kann von einfacher Natur, z. B. eine Zahl, eine Zeichenkette oder ein spezieller Datenspeicher sein, etwa eine Liste, eine (math.) Menge oder auch eine Tabelle. Objekte können andererseits auch Repräsentationen von Lehrpersonen, Studenten, TU-Institutionen oder sogar von Arbeitsabläufen sein.

Instanzen, Instanzvariablen, Methoden

Unabhängig von seiner Natur hat jedes Objekt (wie im realen Leben auch) Eigenschaften und Fähigkeiten. Eigenschaften charakterisieren ein Objekt näher, sie machen es individuell, Eigenschaften werden in so genannten Instanzvariablen festgehalten. Ein kurzes Beispiel: Objekte, die Institute darstellen, gehören der Kategorie oder Klasse „Organisation“ an. Ein konkretes Objekt der Klasse „Organisation“ – oft auch als Instanz bezeichnet – kann beispielsweise nach seiner genauen Institutsbezeichnung, nach seinem Vorstand, sowie nach seinen Mitarbeitern gefragt werden. Diese und ähnliche Angaben sind individuelle „Eigenschaften“ eines Institutes. Und Objekte haben Fähigkeiten, abgebildet als Methoden, die von anderen Objekten durch Senden von „messages“ aufgerufen bzw. genutzt werden können.

Kapselung

Jedes Smalltalk/GemStone Objekt stellt, mitsamt seinen Instanzvariablen und Methoden, eine Art Kapsel dar. Der Kapselinhalt ist in der Regel „Privatsache“: Variablen sind von außen entweder überhaupt nicht oder nur über bestimmte, veröffentlichte Methoden manipulierbar, und auch die tatsächliche Implementierung einer Methode bleibt i.d.R. vor der Außenwelt verborgen. Kurz: um Dienstleistungen eines Objektes nutzen können, muss der anfragende „Kunde“ (z. B. ein anderes Objekt) lediglich die Nachrichten („messages“) des „Dienstleisters“ kennen, nicht aber interne Details über die tatsächliche Ausführung.

Welche Vorteile hat das „Kapsel“-Prinzip? Man kann, wegen der engen logischen Verbindung zwischen Nutzdaten und den darauf anwendbaren Funktionen leichter dafür sorgen, dass ein Objekt zu jeder Zeit richtige und vollständige Daten enthält. Solche Mechanismen findet man in der GemStone/S Datenbank z. B. immer dort, wo neue Objekte erzeugt werden. Ein Objekt beispielweise, welches einen Studenten/eine Studentin repräsentieren soll, hat neben Vor- und Zuname typischerweise auch eine Matrikelnummer und eine Studienkennzahl. Inkorrekt wäre, wenn in der Datenbank plötzlich studentische Einträge ohne Matrikelnummer auftauchten. Durch geschicktes Design kann man Methoden schreiben, die gleichzeitig alle persönlichen Studentendaten setzen und

diese auf Plausibilität hin überprüfen. Ein Anwender braucht sich auch hier wieder keine Gedanken über die Datenkonsistenz machen.

Ein weiterer Vorteil betrifft laufende Ver- bzw. Ausbesserungsarbeiten. Methoden werden von außen durch „messages“ aufgerufen, und liefern i.d.R. definierte Ergebnisse. Bleibt die Nachricht und die Art des erwartbaren Resultats gleich, kann dahinter jede erdenkliche Prozedur ablaufen. Findet ein Programmierer/eine Programmiererin bessere, z. B. schnellere Algorithmen, so kann er/sie diese während des Betriebs ändern, er/sie lässt ja die Schnittstelle nach außen unverändert. Üblicherweise erfüllen Methoden in Smalltalk immer nur relativ kleine Teilaufgaben, der Code bleibt oft überschaubar, sodass sich dadurch manchmal Software-Fehler recht rasch beheben lassen.

Vererbung

Vererbung ist weiteres, wichtiges (manchmal auch überschätztes) Merkmal objektorientierter Software-Systeme. Gleichartige Objekte werden in Klassen eingeteilt. Jede Klasse definiert Instanzvariablen und die verfügbaren Methoden, das Verhaltensrepertoire einer Klasse von Objekten sozusagen. Normalerweise müssen Methoden nach und nach ausprogrammiert werden, objektorientierte Programmiersprachen bieten darüber hinaus an, bereits fertige und anwendbare Methoden zu erben und damit wieder zu verwenden.

Vererbung basiert auf Aussagen wie: „Eine SmallInteger ist ein Spezialfall einer Integer Zahl, eine Integer Zahl ist ein Spezialfall einer Zahl, eine Zahl ist eine Art Größenordnung und eine Größenordnung ist eine Art Objekt.“ Dieser Zusammenhang kann sehr gut auch graphisch, als hierarchische Struktur innerhalb einer Smalltalk/GemStone Klassenbibliothek dargestellt werden:

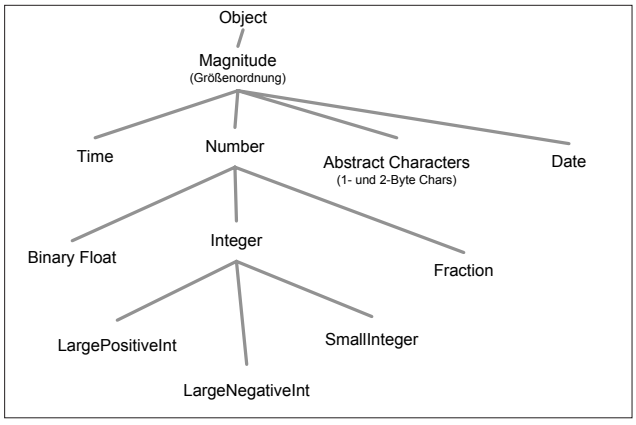


Abbildung 1 - Klassenhierarchien

Man sagt auch: die Klasse „SmallInteger“ ist eine Unterklasse (engl. subclass) von „Integer“. Oder umgekehrt: die Klasse „Integer“ ist die Ober- oder Superklasse (engl. superclass) von „SmallInteger“.

Eine SmallInteger Zahl kennt nicht nur eigene Methoden und Instanzvariablen, sondern auch die ihrer direkten Superklasse, die der Klasse Number usw., d. h. sie erbt Funktionalität. Der größte Vorteil des Vererbungsprinzips ist: eine funktionierende Methode kann auf diese Weise weiter bzw. wieder verwendet werden, was den Entwicklungsaufwand beträchtlich reduzieren kann.

Polymorphismus

Was ist Polymorphismus im Bereich Programmierung? Das deutsche Wikipedia-Portal liefert folgende Begriffserklärung [4]: *„Polymorphie (griechisch, ‘Vielgestaltigkeit’) ist ein Konzept der Programmierung, das es erlaubt, einem Wert oder einem Namen (z. B. einer Variablen) mehrere Datentypen zuzuordnen.“* Innerhalb der GemStone/S Datenbank findet man gelegentlich mehrere, gleichnamige Methodenaufrufe. Ein Beispiel:

```
20 + 4. 24
```

In Smalltalk ist „+“ eine Nachricht und das „+“-Zeichen ist das Symbol für eine Methode, die 20 und 4 addiert und anschließend das Resultat ausgibt. Die Addition lässt sich also bei Datenobjekten der Kategorie „SmallInteger“ anwenden. Bemerkenswerterweise wird dieses Nachrichtensymbol auch noch an anderer Stelle verwendet.

Beispiel 1:

```
'a' class. String
'a' + 'b'. ab
'a' class whichClassIncludesSelector: '+'.
SequenceableCollection
```

Vorhin sind zwei ganze Zahlen addiert worden, in Beispiel 1 sind die Summanden aber Zeichenketten, die jeweils ein Zeichen lang sind. In Zeile 2 sieht man, dass man mit Hilfe der Nachricht „+“ sogar „Nicht-Zahlen“ verknüpfen kann. Aber die Addition zweier Zahlen und das Aneinanderreihen zweier Zeichenketten sind doch unterschiedliche Vorgänge, oder? Von außen betrachtet sieht es jedenfalls so aus, als würde die Methode hinter dem „+“ Symbol ihre Funktion, ihr „Aussehen“ den Summanden anpassen. Kurz: die Methode wirkt sozusagen „polymorph“.

Methoden verändern auch noch an anderen Stellen ihr Aussehen: im Abschnitt „Vererbung“ ist erwähnt worden, dass Objektsklassen in hierarchischen Beziehungen stehen und dass sog. „subclasses“ meist Spezialfälle einer „superclass“ sind. Um es mit einem alltäglichen Beispiel zu illustrieren: Hunde und Katzen sind beliebte Haustiere. Haustiere können Laute von sich geben, der „Spezialfall“ Hund bellt, während eine Katze z. B. schnurren kann. In objektorientierter Hinsicht muss die Methode „Laut geben“ also von den Unterklassen „Hund“ bzw. „Katze“ weiter spezialisiert werden. Sofern die Oberklasse „Haustier“ schon eine Methode „Laut geben“ kennt und vererben kann, wird diese in der Unterklasse – z. B. von einer Katze – u.U. sogar vollständig neu geschrieben. Im Fachjargon nennt man den Vorgang des Neuschreibens einer Methode „overriding“ [5].

Smalltalk

Smalltalk ist die Programmier- bzw. Abfragesprache dieser Datenbank, Smalltalk zählt sicherlich zu den „Ursprachen” der Informatik. Inspiriert von Simula-, LISP- und Logo-Konzepten entstand zwischen 1971 und 1975, unter der Leitung von Alan Kay et al., bei Xerox PARC, erstmals eine vollständige Smalltalk-Entwicklungsumgebung, die schon damals ein Windowing-System anbot. Jahre später schrieb Adele Goldberg ihre legendäre Smalltalk-Bibel „Smalltalk-80, The Interactive Programming Environment“.

Semantik und Syntax

Die Smalltalk Sprachsyntax [6] ist im Grunde leicht erlernbar. Die Sprache ist „case sensitive“, es wird zwischen Groß- und Kleinbuchstaben unterschieden. Jede Programmzeile endet mit einem Punkt, jeder Aufruf folgt einem gleichbleibenden Schema:

```
Objekt Methode [Parameter].
```

Einem Objekt können auch mehrere „Aufträge“ hintereinander, getrennt durch Strichpunkte, geschickt werden:

```
Objekt Methode1 [Parameter1];
  Methode2 [Parameter2];
  Methode3 [Parameter3].
```

Auch „geschachtelte“ Aufrufe sind möglich:

```
Objekt1 Methode1 ( Objekt2 Methode2 ).
```

Der Klammerausdruck erzeugt als Resultat ein Objekt, welches seinerseits zu einem Parameter für Methode „Methode1“ wird.

Wichtig ist, dass man das Prinzip „Objektorientierung“ und damit verbundene Ideen, wie Klassenhierarchien, einigermaßen verstanden hat. Der markanteste Unterschied zu klassischen, prozeduralen Sprachen wie C besteht darin, dass man in Smalltalk geradezu gezwungen wird, sich vor Verfassen einer Codezeile Gedanken zur zugehörigen Klasse zu machen. Oder anders gesagt: in Smalltalk benötigt man Klassen, auch um überhaupt Methoden programmieren zu können. Smalltalk ist damit eine strikt objektorientierte Sprache.

Ein GemStone/S Entwickler muss vor allem wissen, wie neue Klassen deklariert werden. Beispiel 2 zeigt das Erstellen einer neuen Klasse namens „Tier“ innerhalb einer GemStone/S Datenbank.

Beispiel 2:

```
Object subclass: 'Tier'
 instVarNames: #( gattung art nahrung lebensraum )
 inDictionary: UserGlobals.
```

Die Klasse „Tier“ wird erzeugt, indem man der „obersten“ Oberklasse „Object“ den Auftrag gibt, eine neue Unterklasse zu bilden (Schlüsselwort „subclass:“, gefolgt vom gewünschten Klassennamen), diese mit entsprechenden Instanzvariablen auszustatten (Schlüsselwort „instVarNames:“, gefolgt von einem Feld, bestehend aus Variablennamen), und eine Referenz zu dieser neuen Klasse in einer Art „Wörterbuch“ einzutragen, um sie dauerhaft im GemStone/S Speicher („repository“) aufbewahren zu können. Erst jetzt werden Methoden formulierbar, die das „Tier“ zu einem aktiven Objekt werden lassen.

Kann man in diesem – zugegeben trivialen – Beispiel auch Vererbung zeigen? Ja, man kann. In Beispiel 3 wird eine weitere Unterklasse von „Tier“ vereinbart:

Beispiel 3:

```
Tier subclass: 'Haustier'
  instVarNames: #( name besitzer adresse )
  inDictionary: UserGlobals.
Haustier compileAccessingMethodsFor: #( name
  besitzer adresse ).
```

Ein Haustier ist offenbar ein Spezialfall von einem Tier. Warum? Weil es, im Gegensatz zu einem frei lebenden, meist einen Namen trägt, einem Besitzer/einer Besitzerin „gehört“ und sich sogar eine Adresse angeben lässt, wo es „wohnt“. Sobald die Klasse „Haustier“ existiert, kann man konkrete Objekte erzeugen, wie in Beispiel 4.

Beispiel 4:

```
| einHaustier |
einHaustier := Haustier new.
```

Nach seiner generellen Vereinbarung in Zeile 1, wird die Variable „einHaustier“ erst in Zeile 2 zu einem Objekt der Klasse „Haustier“. Auch hier unterscheidet sich Smalltalk wieder von anderen Programmiersprachen, die an ähnlicher Stelle einen Variablennamen samt Typenvereinbarung verlangt hätten (Stichwort „late binding“).

Und wo merkt man Vererbungsmechanismen? Schon in Zeile 2: hier wird der Variablen „einHaustier“ durch das Symbol „:=“ ein Wert, genauer ein Objekt zugewiesen. Das Objekt gehört offenbar der Klasse „Haustier“ an und kann mit der Nachricht „new“ erzeugt werden. Es ist eine Methode, die vorhin (in Beispiel 3) nicht extra angegeben worden ist, die dahinterliegende Methode „new“ muss also geerbt worden sein.

Mit dem Objekt „einHaustier“ kann bereits „gearbeitet“ werden: man kann z. B. dessen Eigenschaften festlegen: Frau Rosalinde soll die Besitzerin einer Katze namens Minka sein und – falls Minka entlaufen sein sollte – soll man feststellen können, wo die Katze eigentlich zuhause ist. In Beispiel 3 sind „accessing methods“ erstellt worden, die nun genutzt werden können, z. B. die Methoden „name:“, „besitzer:“ usw.

Beispiel 5:

```
einHaustier name: 'Minka';
  besitzer: 'Rosalinde';
  adresse: 'Gartengasse 12';
  art: 'Felis domestica';
  gattung: 'Felis'.
```

Beispiel 6:

```
einHaustier art.  Felis domestica
einHaustier name.  Minka
```

In Beispiel 5 werden Eigenschaften des konkreten Haustiers festgelegt und in Beispiel 6 abgefragt. Ab jetzt könnte man die Klassen „Tier“ und „Haustier“ weiter „modellieren“ und beispielsweise mit weiteren Methoden ausstatten, die deren Verhalten festlegen.

Was fällt noch auf? Obwohl nicht eigens definiert, versteht das Haustier-Objekt auch die Abfrage „art“, ein weiteres Beispiel für Vererbung. Methodennamen haben imperativen Charakter, sie muten wie Befehle an. Durch geschickte Namenswahl kann man oft den Sinn und Zweck einer Methode erraten, ohne den Quellcode lesen zu müssen.

Smalltalk-Entwicklungsumgebungen

Wie sieht eine typische Smalltalk-IDE aus? Um zu einer VisualWorks IDE zu gelangen, wird eine Virtuelle Maschine („visual“) gestartet. Als Parameter wird dem Aufruf der Name eines sog. „Images“ („workingWithGemstone.im“) mitgegeben, in dem sich die komplette Smalltalk-Klassenbibliothek sowie selbst geschriebene Anwendungen befinden können:

```
mi@gandalf:~> $VISUALWORKS/bin/linux86/visual
$HOME/workingWithGemstone.im
```

Zunächst wird nach diesem Aufruf das sog. VisualWorks-„Transcript“-Window geöffnet. Es ist jenes Fens-

ter, über welches in erster Linie Smalltalk-Systemmeldungen oder Ergebnisse ausgegeben werden. Das weitaus wichtigere Werkzeug ist aber der sog. Smalltalk-Klassen-Browser, mit dem man Zugriff auf alle vorhandenen Klassendefinitionen und deren Methoden erhält:

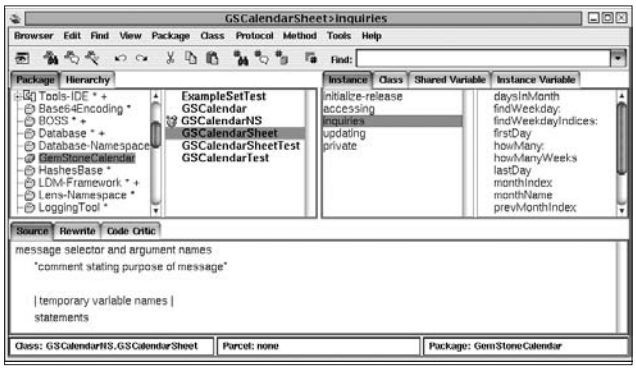


Abbildung 2 - VisualWorks Klassen-Browser

Im Grunde ist der „GemStone/S Object Server“ nichts anderes als eine Smalltalk-IDE, ergänzt mit „Multiuser“-Funktionen.

GemStone/S an der TU Wien

Datenstrukturen

In der GemStone/S Datenbank am ZID gibt es mehr als 95 Klassendefinitionen und mehr als 1 Million „lebende“ Objekte. Mit „lebend“ sind Objekte gemeint, die aktuelle Daten enthalten und die wiederauffindbar sind, d. h. referenzierbar sind. Die Klassennamen sind meistens selbsterklärend und beziehen sich i.d.R. auf Begriffe oder Arbeitsabläufe innerhalb des ZID oder der TU Wien. Relativ häufig verwendet wird die Klasse „Person“. Im Folgenden soll anhand dieser Klasse beispielhaft gezeigt werden, auf welche Art diverse Klassen in Beziehung zueinander stehen.

Statische, vernetzte Klassen- bzw. Objektsbeziehungen

Beispiel 7 zeigt die Definition der Klasse „Person“. Die Vereinbarung unterscheidet sich von der in Beispiel 2 gezeigten in einigen Punkten.

Beispiel 7:

```
Object subclass: 'Person'
  instVarNames: #( title firstName lastName
                   licenceCache accounts roles
                   birthday )
  classVars: #( NameTree RemovalLog
                SpecialUserMap UserMap )
  classInstVars: #()
  poolDictionaries: #[]
  inDictionary: IU_Infoproject
  constraints: #[ #[ #title, String],
                  #[ #firstName, String],
                  #[ #lastName, String],
                  #[ #licenceCache, LicenceSet],
                  #[ #accounts, AccountList],
                  #[ #roles, Collection],
                  #[ #birthday, Integer] ]
  instancesInvariant: false
  isModifiable: false
```

Als Erstes fällt auf: die Klasse „Person“ kennt sog. Klassenvariablen (Abschnitt „classVars“). Es sind Speicher, auf die alle Personenobjekte zumindest lesend zugreifen können. Auf diese Weise kann man Informationen „global“ verteilen. Ein weiterer Unterschied zu Beispiel 2 ist: in dieser Klasse ist erstmals festgelegt, welcher Art die Instanzvariablen sein dürfen. Diese Einschränkungen stehen im Abschnitt „constraints“, einer GemStone/S Erweiterung. Eine Zeile daraus, z. B.

```
#[ #licenceCache, LicenceSet],
```

besagt, dass eine Person (neben Titel, Vor- und Zuname usw.) auch einen sog. „licenceCache“ hat, der vom Typ „LicenceSet“ sein muss. Was ist in so einem „Lizenz Cache“ verzeichnet? Die Antwort lautet: alle, von einer Person gekauften bzw. gemieteten Campus- oder Studenten-Softwarelizenzen. Auch ein „LicenceSet“ ist eine Objektsklasse, die ihrerseits nur Elemente der Klasse „Licence“ enthalten darf.

Umgangsprachlich kann man diese Art von Klassen- bzw. Objektsbeziehungen auch so formulieren: eine Person kann einen Titel tragen, sie hat einen oder mehrere Vor- bzw. einen Zunamen sowie einen Geburtstag, sie kann Software-Lizenzen und/oder TU-Zugänge („accounts“) gemietet haben und sie kann entweder TU-, Drittmittel- oder Projektmitarbeiter/in sein (siehe „roles“). Damit werden vor allem statische, zeitlich praktisch unveränderliche Beziehungen zwischen Objekten beschrieben. Je weiter man diesen „ist“- und „hat“-Beziehungen folgt, umso netzartiger werden die Zusammenhänge zwischen den GemStone/S Objekten. Georg Gollmann hat diesen Umstand anlässlich einer „European Smalltalk User Group“ Konferenz 2004 folgendermaßen graphisch dargestellt:

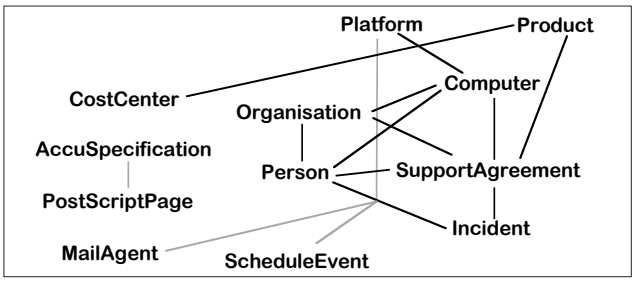


Abbildung 3 - Klassenbeziehungen nach Georg Gollmann

Dynamische Beziehungen

Innerhalb der GemStone/S Datenbank gibt es auch „dynamische“ Beziehungen. Das sind im Wesentlichen temporäre Zusammenhänge, die während bestimmter Arbeitsabläufe entstehen. Wir wissen: Objekte erfüllen immer relativ kleine Aufgaben, deren Dienste können aber von anderen Objekten genutzt werden. Dienste eines Objektes können von außen durch „messages“ genutzt werden. Und um einen größeren Auftrag erledigen zu können, müssen in der GemStone/S Datenbank oft mehrere Objekte miteinander kommunizieren, kurz: „messages“ zwischen unterschiedlichen GemStone/S Objekten hin- und hergeschickt werden. Im Fachjargon heißt diese Art von Kommunikation bzw. das Aufrufen von Methoden auch „message passing“.

Abteilungsinterne Dienste

Bezüglich der Aufgaben, die die GemStone/S Datenbank an der TU Wien übernimmt, kann man sagen: ihr Aufgabenbereich lässt sich unterteilen in abteilungsinterne und in TU/ZID-weit angebotene Dienste.

Die GemStone/S Datenbank ist für Abteilung Standardsoftware (STS) das zentrale Werkzeug. Mit ihrer Hilfe wird der Vertrieb von Campus- bzw. Studenten-Software-Lizenzen abgewickelt. Eine öffentliche Schnittstelle der Datenbank ist übrigens im Lehrmittelzentrum (LMZ), gleich neben der Kassa, zu sehen. Abbildung 4 zeigt die Benutzeroberfläche zum GemStone/S Verkaufssystem. Mit einem Handscanner liest der/die Verkäufer/in die Matrikelnummer vom Studentenausweis ein, der Verkaufs-PC kontaktiert die GemStone/S Datenbank, die die Bezugsberechtigung prüft und gleichzeitig Buch führt über die getätigten Einkäufe des/der Studenten/in.

Abbildung 4 - GemStone/S Verkaufssystem

In dieser Datenbank sind alle STS-Kunden und Produkte verzeichnet, sie ist ein internes Dokumentationssystem, Fallverfolgungssystem für Wartungsvertrags-Unterstützungsfälle und Abrechnungssystem in einem, sie erfasst das STS-Inventar und bildet das STS-eigene Bestellwesen ab. Die Funktion des Bestellwesens ist seit 2004 im Einsatz. Es ermöglicht STS-Mitarbeiter/innen das problemlose, unbürokratische Absetzen von Bestellungen für Neuerwerbungen, Ersatzteile, Seminare, Bücher etc. und ist mit dem nachträglich eingeführten SAP-System gekoppelt. Abbildung 5 zeigt den „Workflow“ eines Bestellvorgangs.

Eingetroffene Waren, die den Wareneingang erfolgreich passiert haben, werden im GemStone/S Bestellsystem „abgelegt“. „Abgelegt“ bedeutet i.d.R., dass diese Anlage-, Verbrauchsgüter etc. von der Datenbank automatisch ins eigene Inventarsystem übernommen werden. Mit dieser Funktion verfügt der STS-Leiter, Herr DI Albert Blauensteiner, stets über eine tagesaktuelle Budgetübersicht und kann Abschreibekalkulationen und Anlagenbewertungen praktisch auf „Knopfdruck“ durchführen lassen. Er ist damit zu jeder Zeit über aktuelle Kostenverläufe informiert und hat damit die Basis für präzise budgetäre Planungen.

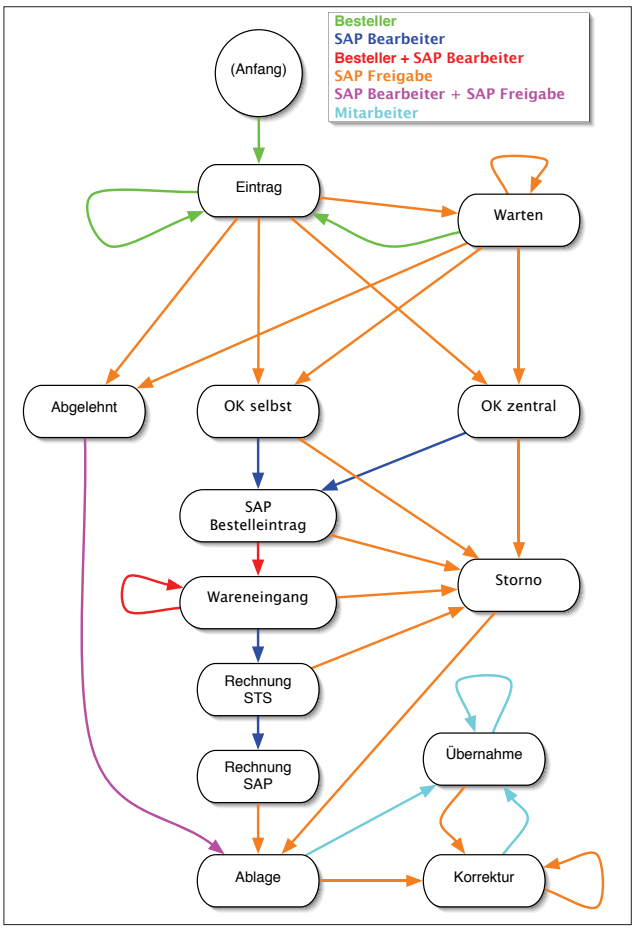


Abbildung 5 - Bestellwesen, „Workflow“

Benutzerschnittstelle & Entwicklerwerkzeuge

Die GemStone/S Datenbank ist nicht nur ein reiner Datenspeicher, sie ist – wie vorhin gezeigt – ein wichtiges, zentrales Verwaltungsinstrument und Entwicklungssystem zugleich. Ihre primäre Schnittstelle ist ein Web-Server, der – natürlich auch in Smalltalk geschrieben – integrierter Bestandteil der Datenbank ist. Der Zugriff ist damit für die meisten Benutzer mittels Web-Browser möglich, wobei die Verbindungen durch SSL („Secure Socket Layer“) geschützt werden. Es sind zwei unterschiedliche Zugänge via Web-Browser möglich: die eine Variante erfordert einen expliziten Datenbank-Login, wodurch eine so genannte „Private Session“ eröffnet wird. „Private Sessions“ sind i.d.R. Mitarbeiter/innen der Abteilung STS vorbehalten. Zugänge ohne Login, die mehr oder weniger öffentlich sein sollen, können auch anders verifiziert werden, z. B. durch das TU-Passwort. Mit dem TU-Passwort kann man beispielsweise eigenen „White Pages“-Einträgen weitere Informationen hinzufügen, eine Funktion, die von der GemStone/S Datenbank bereit gestellt wird. Jeder Benutzer sieht beim Zugriff eine andere GemStone/S Oberfläche, deren Aussehen hängt von der jeweiligen Funktion und Rolle des/der Mitarbeiter/in ab. Abbildung 6 zeigt das sog. „Graphical User Interface (GUI)“ eines GemStone-Entwicklers.

Die eigentlichen Entwicklungswerkzeuge sind über die „Developers:“-Zeile erreichbar. Es gibt einen „Workspace“, in dem Smalltalk-Code quasi „ausprobiert“ werden kann. Die in den vorherigen Beispielen gezeigten Abfra-

gen, Berechnungen und Klassenvereinbarungen können kopiert und direkt im „Workspace“ eingetragen werden.

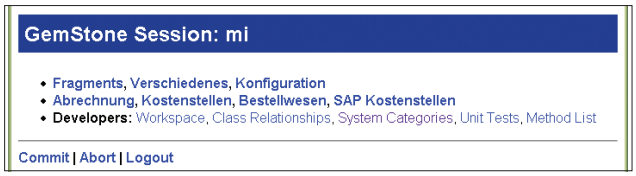


Abbildung 6 - Entwickler GUI

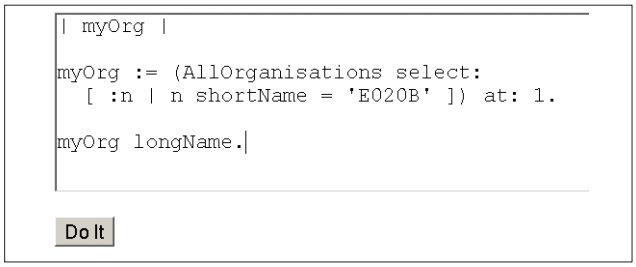


Abbildung 7 - GemStone/S Workspace

Die Code-Zeilen werden praktisch zur Laufzeit übersetzt bzw. interpretiert, man kann dadurch leicht und rasch Smalltalk-Code auf dessen Brauchbarkeit hin untersuchen, bevor dieser Teil einer Methode wird. Das Ergebnis eines beliebigen Aufrufs ist – wie in Smalltalk üblich – selbst wieder ein Objekt und wird in einem separatem Browser-Fenster dargestellt.

Hinter der Verknüpfung „Class Relationships“ kann ein Entwickler feststellen, welche statischen Beziehungen zwischen ausgewählten GemStone/S-Klassen bestehen. Möchte man z. B. wissen, mit welchen anderen Klassen die Klasse „Person“ in Beziehung steht, so erhält man folgende graphische Darstellung:

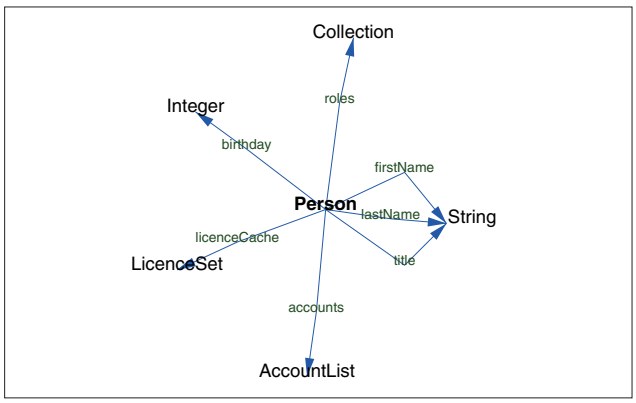


Abbildung 8 - Statische Klassenbeziehungen

Hinter der Verknüpfung „System Categories“ steckt das vermutlich wichtigste Werkzeug: der GemStone/S Klassen-Browser. Er zeigt alle existierenden Klassendefinitionen, welche unterteilt sind in GemStone/S Kernel-Klassen, in Klassen, die entweder STS-spezifische Funktionen (Kategorie „IU_Infoproject“) oder TU/ZID-weite Dienste bereitstellen (Kategorie „ZID DB“). Außerdem gibt es Klassen, die die eigentlichen Daten darstellen. Daten können in „Arrays“, in „Sets“ oder in „Bags“ u.ä. dauerhaft gespeichert werden, die Klassendefinitionen dazu findet man in der Kategorie „Collections“. Eine praktische Zusatzfunktion bietet die Detailansicht einer Klasse:

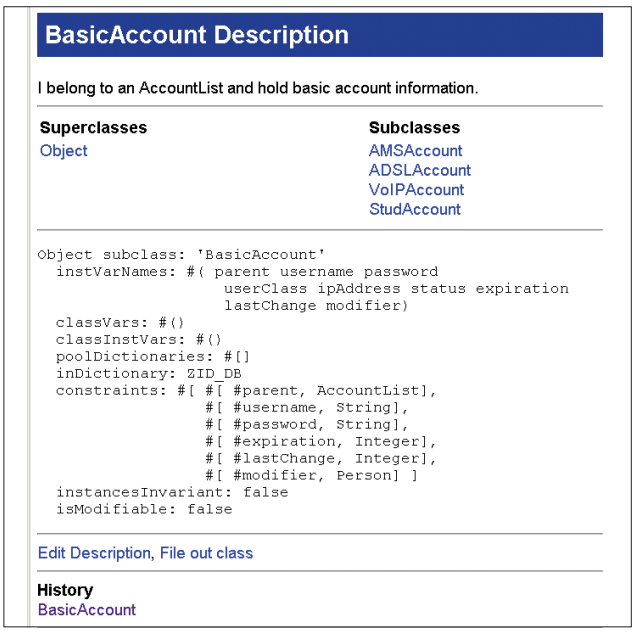


Abbildung 9 - Klassen-Detailansicht

Man kann sich die sog. Ober- bzw. Unterklassen („superclasses“ bzw. „subclasses“) auflisten lassen und bekommt dadurch u.a. Hinweise, welche zusätzlichen Funktionen eine Klasse – wegen Vererbung – sonst noch erfüllen kann. Über den Menüpunkt „Method List“ kann ein/e Entwickler/in nach Methoden suchen. Er/Sie kann damit feststellen, welche Klasse eine bestimmte Methode implementiert bzw. wie oft der Methodenname im System ingesamt vorkommt (siehe Abschnitt „Polymorphismus“).

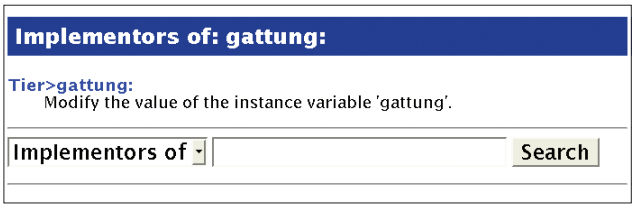


Abbildung 10 - GemStone/S Methodensuche

GemStone/S Entwicklungszyklen lehnen sich stark an Empfehlungen der „eXtreme Programming (XP)“ Fachleute [9] an. Eine dieser Empfehlungen lautet: „*Bevor Du tatsächlich Code-Zeilen schreibst, nimm die vorgegebene ‘User Story’ und entwickle daraus zunächst nur Tests. Diese Tests sollen genau das tun, was der spätere Benutzer tun möchte/tun will. Beginne erst danach mit dem eigentlichen Entwicklunsprozess, die Software wird fertig sein, wenn sie alle Tests erfolgreich absolviert.*“

Die GemStone/S Datenbank enthält eine Klasse „TestCase“, mit der diverse Testfälle formuliert werden können. Erfüllt eine Datenbank-Teilfunktion alle geforderten Aufgaben, so sieht das so aus:

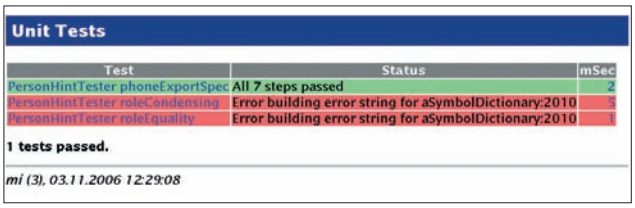


Abbildung 11 - Unit Tests in GemStone/S

ZID/TU-weite Dienste

Zu den wichtigsten TU-weiten GemStone/S Diensten gehört sicherlich die ZID Personendatenbank. Diese ist – wie bereits erwähnt – eine Teilfunktion und übernimmt eine Art Verteilerrolle: die eigentlichen Daten werden in der zentralen Verwaltung (z. B. Namen von TU-Angestellten und Studierenden) oder in der Telefonanlage (z. B. Nebenstellendurchwahl) generiert, anschließend von der GemStone/S Datenbank importiert und an die „White Pages“ weitergegeben. Die genaue Funktion der Personendatenbank erläutert Herr DI Georg Gollmann in seinem ZIDline-Artikel vom Oktober 2002 [2].

Darüber hinaus unterstützt die Datenbank auch die Verwaltung von ZID-Accounts und sie ist gleichzeitig Authentifizierungsserver für eine Reihe von Web-Anwendungen an der TU Wien. Anwender dieser GemStone/S Funktion ersparen sich damit das Entwickeln eigener, aufwendiger Authenfizierungsmechanismen. Detaillierte Informationen dazu finden sich in [7] und [8].

Resumee

Smalltalk

Smalltalk ist eine flexible, moderne Programmiersprache. Sie wird heute in Österreich kaum verwendet oder gelehrt, was verwunderlich ist, wenn man bedenkt, dass einige Java-Konzepte aus der Smalltalk-Welt stammen, so zu Beispiel das Konzept der Virtuellen Maschinen. Heute gibt es Smalltalk-IDE für UNIX, MacOS, WinXP und natürlich Linux, eine Anwendung entwickelt unter WinXP läuft somit auch unter Linux.

Hinsichtlich der Verbreitung stellt sich die Situation in Deutschland ähnlich dar, an folgenden Stellen wird erfolgreich mit Smalltalk/VisualWorks gearbeitet: die Firma Carl Zeiss in Oberkochen entwickelt damit ihre Messsoftware „Calypso“ [13]. Am Standort Dresden steuert der Intel-Konkurrent AMD „Wafer“ Fertigungsprozesse teilweise mit Smalltalk-basierender Software. Der Heise-Verlag hat in den letzten Jahren einige Artikel zu Smalltalk veröffentlicht [14].

Wer Smalltalk lernen und die Geheimnisse dieser Sprache kennenlernen möchte, der kann entweder bei Georg Heeg ein Seminar besuchen [16] oder sich ein Buch kaufen oder ausborgen. An der TU Bibliothek verfügbar sind z. B. [10, 11].

Die unbestrittene Smalltalk-Kompetenz an der TU Wien ist Herr DI Georg Gollmann (DW 42022, E-Mail: gollmann@zid.tuwien.ac.at).

Für Lehr- und Lernzwecke gibt es glücklicherweise frei verfügbare Smalltalk-Entwicklungswerkzeuge (mit entsprechender Dokumentation), z. B. [18, 19]. Diese Smalltalk-IDE bieten alles, was Entwickler/innen brauchen können: Klassen-Browser, „Workspaces“, Debugger und gute Dokumentation, besonders für VisualWorks.

GemStone/S

Der „GemStone/S Object Server“ kann ebenfalls – z. B. zu Evaluierungszwecken oder zur persönlichen Weiterbildung – frei bezogen werden: http://www.gemstone.com/products/smalltalk/.

Die Dokumentation ist nicht Bestandteil des „Object Servers“, sie muss eigens geholt werden. Der „Object Server“ wird üblicherweise mit einer 30-Tage-Demolizenz geliefert, eine namentliche Registierung ist erforderlich.

Die GemStone/S Datenbank ist ein überaus flexibles, stabiles Produkt, das heute von sehr großen Unternehmen, wie z. B. der „Orient Overseas Container Line“ [20] verwendet wird. Die Flexibilität wird – wie gezeigt – durch objektorientierte Betrachtungsweisen erreicht. An der TU Wien hat sie jedenfalls bewiesen, dass sie auch nach 9 Jahren noch immer erweiterbar ist [2]. Eine gutes Zeichen für zukünftige Aufgaben.

Literaturangaben

[1] Klasek, Johann: White Pages Service, ZIDline Nr. 7, Oktober 2002, S. 11ff.

[2] Gollmann, Georg: ZID Personendatenbank, ZIDline Nr. 7, Oktober 2002, S. 16f.

[3] GemStone Systems, 1260 NW Waterhouse Ave., Suite 200, Beaverton, Oregon 97006, http://www.gemstone.com

[4] Wikipedia: Polymorphie (Programmierung), Oktober 2006, http://de.wikipedia.org/wiki/Polymorphie_%28Programmierung%29

[5] Hunt, John: Smalltalk and Objectorientation, An Introduction, Springer Verlag, Berlin 1997, ISBN: 3540761152

[6] GemStone/S Programming Guide, Version 6.0, GemStone Systems, December 2001

[7] Gollmann, Georg: Authenfizierungsservice, ZIDline Nr. 7, Oktober 2002, S. 35f.

[8] Gollmann, Georg: Dateninfrastruktur, http://sts.tuwien.ac.at/go/Dateninfrastruktur.html

[9] Wells, Don: Extreme Programming: A gentle introduction. http://www.extremeprogramming.org/

[10] Johannes Brauer: „Grundkurs Smalltalk, Objektorientierung von Anfang an“, Vieweg Verlag, Wiesbaden: 2004 (2., verb. Aufl.), ISBN: 3-528-15818-2

[11] Beck, Kent: Kent Beck’s guide to better smalltalk, Cambridge Univ. Pr., Cambridge: 1999 (1. Aufl.)

[12] Smalltalk Industry Council (STIC): Why Smalltalk, http://www.whysmalltalk.com/

[13] http://www.zeiss.de/Calypso-fuer-alle

[14] http://www.heise.de/ct/inhverz/search.shtml?T=Smalltalk&Suchen=suchen

[15] Vorlesungen u.a. zum Thema Smalltalk: http://brauer.nordakademie.de/sites/main.html

[16] Seminare Fa. Georg Heeg Köthen bzw. Dortmund: http://www.heeg.de/

[17] Die Smalltalk-„Online“-Referenz ist: „Why Smalltalk“, http://www.whysmalltalk.com/

[18] VisualWorks 7.x von CinCom: http://www.cincomsmalltalk.com/userblogs/cincom/blogView

[19] Squeak: http://www.squeak.org/

[20] http://www.oocl.com/, http://www.gemstone.com/pdf/OOCL_SuccessStory.pdf

Honeynet-Projekt

Christopher Krügel und Engin Kirda
Secure Systems Lab
chris@seclab.tuwien.ac.at, ek@seclab.tuwien.ac.at

Ein Honeynet hat die Aufgabe, Angriffe auf ein Netzwerk zu protokollieren. Dazu wird ein unbenutzter IP-Adressbereich im Netzwerk reserviert und überwacht. Alle Verbindungen zu Maschinen in diesem Bereich sind automatisch verdächtig und können genauer analysiert werden. Das dient zum Beispiel dazu, Netzwerkscanner zu identifizieren oder den Code (die Payload) von Computerwürmern für eine spätere Auswertung zu sammeln.

Je größer der überwachte Adressbereich ist (das heißt, je mehr IP-Adressen Teil des Honeynets sind), desto mehr und genauere Informationen können gesammelt werden. Für unsere Forschung im Gebiet der Analyse von bösartigem Code (Malware) haben wir zusammen mit dem ZID ein Honeynet auf der TU Wien installiert, das mehr als vier tausend IP-Adressen umfasst. Dieses Honeynet registriert im Durchschnitt rund 250.000 Verbindungen täglich. Allein im Oktober 2006 haben wir 180 verschiedene Malware Samples (Würmer) gesammelt und 24 Netzwerkscanner innerhalb der TU Wien an die Security Abteilung des ZID gemeldet.

Einleitung

Ein Honeynet ist eine Menge von Maschinen (so genannte Honeypots), welche die Aufgabe haben, Angriffe auf ein Netzwerk aufzuzeichnen. Die Grundidee eines Honeypots ist es, eine Menge von Netzwerkdiensten anzubieten, die legitimen Benutzern unbekannt sind und die keine Funktion im operativen Betrieb des Netzwerk erfüllen. Nachdem die Dienste unbekannt sind, können sie nur dann gefunden werden, wenn ein Angreifer das gesamte Netzwerk nach bestimmten Diensten absucht. Typischerweise geschieht dies mittels eines Portscanners, der Anfragen an alle Rechner im Netz sendet. Falls ein Dienst von einer Maschine angeboten wird, beantwortet diese die Anfrage entsprechend und signalisiert dadurch dem Angreifer die Verfügbarkeit eines Services. Nachdem der Angreifer nicht weiß, welches Service zu den operativen Rechnern und welches zu einem Honeypot gehört, werden Attacken normalerweise gegen alle entdeckten Dienste gestartet. Ein Honeypot sollte jedoch niemals legitime Anfragen erhalten. Daher ist jede Verbindung zu einem der Dienste eines Honeypots automatisch verdächtig.

Man unterscheidet üblicherweise zwischen zwei verschiedenen Arten von Honeypots, low-interaction und high-interaction Honeypots [1]. Ein low-interaction Honeypot stellt keine wirkliche Implementierung eines Netzwerkdienstes zur Verfügung, sondern emuliert nur einen Teil der Funktionalität. Im einfachsten Fall wird auf jede Anfrage mit einer standardisierten Antwort reagiert (z. B bei honeyd [2]). Andere low-interaction Honeypots realisieren einen größeren Teil der Funktionalität und erlauben es, mehrere Pakete mit einem Angreifer auszutauschen (z. B. nepenthes [3] oder honeytrap [4]). Dies hat den großen Vorteil, dass bei solchen Honeypots mehr Daten vom Angreifer gesendet werden, die dann gespeichert und analysiert werden können. So kann man zum Beispiel die Payload von neuer Malware (wie z. B. von Würmern) sammeln, die sich automatisch im Netzwerk verbreiten.

Bei einem high-interaction Honeypot wird das Netzwerkservice tatsächlich vollständig implementiert und läuft typischerweise auch direkt auf einem richtigen Server. In diesem Fall lassen sich natürlich die meisten Informationen sammeln. Das Problem mit high-interaction Honeypots ist jedoch, dass es aufwendig ist, diese zu installieren und zu warten. So muss man zum Beispiel nach

jedem erfolgreichen Angriff gegen einen Netzwerkdienst den Rechner von den Auswirkungen dieser Attacke säubern, typischerweise durch eine Neuinstallation des Betriebssystems. Außerdem lassen sich auf einer physikalischen Maschine oft nur wenige high-interaction Honeypots parallel unterbringen (z. B. mittels VMware [5]), während ein einzelner Rechner oft tausende low-interaction Honeypots laufen lassen kann. Dies ist wichtig, wenn man bedenkt, dass von einem Honeynet oft mehrere tausend IP-Adressen abgedeckt werden müssen.

Honeynet-Infrastruktur an der TU Wien

Nachdem die Dienste eines Honeypots durch einen Angreifer gefunden werden sollen, ist es von Vorteil, möglichst viele Honeypots in einem Honeynet zusammenzufassen. Vor allem wenn ein Angreifer (oder ein Computerwurm) zufällig ausgewählte IP-Adressen mit einem Portscan untersucht, erhöht sich klarerweise die Chance, dass ein Honeypot darunter ist, wenn mehr Adressen von dem Honeynet abgedeckt werden. Daher war es uns sehr wichtig, vom ZID einen möglichst großen, unbenutzten Adressbereich zur Verfügung gestellt zu bekommen. Der gesamte Traffic zu diesen Adressen sollte dabei zu unserem Honeynet umgeleitet werden. Der ZID hat uns für unser Honeynet großzügigerweise 4.096 IP-Adressen zur Verfügung gestellt, die in 16 Class C Netzwerke mit jeweils 256 Adressen aufgeteilt sind. Das Routing des TU-Netzwerks wurde vom ZID so angepasst, dass alle Pakete an diese Adressen an einen Rechner in unserem Lab weitergeleitet werden, der das Honeynet hostet. Dieser Rechner ist so konfiguriert, dass er auf Anfragen an jede der 4.096 IP-Adressen entsprechend antwortet.

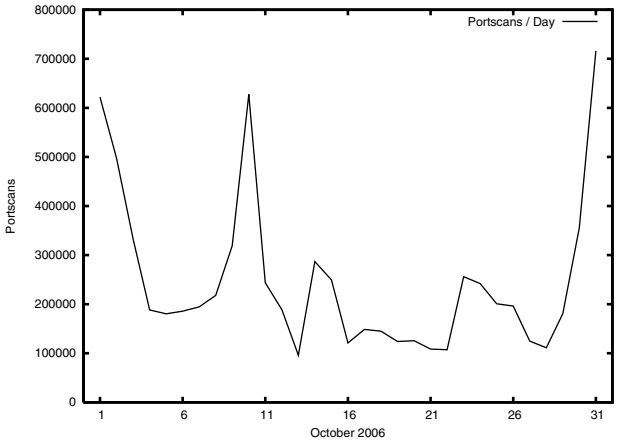
Ein wichtiger Forschungsschwerpunkt bei uns am Secure Systems Lab [6] ist das Studium von Malware (wie Viren und Würmer). Insbesondere untersuchen wir Techniken, um neuartige Formen von bösartigem Code erkennen und klassifizieren zu können. Dazu ist es natürlich notwendig zu wissen, welche Arten von Malware sich zurzeit besonders aktiv im Internet verbreiten. Außerdem benötigen wir real-world Samples, an welchen wir unsere neuen Algorithmen und Techniken ausgiebig testen können. Um einen Einblick in die Verbreitung von Malware zu erhalten und eine repräsentative Menge von Vertretern der verschiedenen Klassen sammeln zu können, eignen sich besonders low-interaction Honeypots. Deshalb haben wir auf unserem Honeynet nepenthes installiert, ein low-interaction Honeypot, der explizit zum Sammeln von Malware, Viren und Trojanern ausgelegt ist.

Ein wichtiger Punkt, der berücksichtigt werden muss, ist, dass das Weiterleiten des Netzwerkverkehrs vom ZID an unser Honeynet vor etwaigen Filtern oder Firewalls erfolgt, damit wir die ganze Breite von bösartigem Traffic überwachen können. Das widerspricht allerdings der Sicherheitspolicy des ZID, die einen Grundschutz für alle Rechner innerhalb des Netzwerks vorsieht [7]. Dieser Grundschutz blockiert automatisch Pakete zu bestimmten Diensten, die sich in der Vergangenheit als besonders anfällig für Angriffe gezeigt haben. Dieser Grundschutz ist natürlich vernünftig, um das Netzwerk vor Angreifern zu schützen. Er stellt allerdings ein Problem für unser Honeynet dar, weil wir ja gerade an diesen Angriffen interessiert sind. Glücklicherweise konnte ein Kompromiss erzielt werden und ein Class C Netzwerk kann jetzt völlig uneingeschränkt vom Internet erreicht werden. Dies erlaubt uns auch, einen Vergleich zu ziehen zwischen jenem Bereich, der vom Grundschutz abgedeckt ist, und jenem, der freigeschaltet ist. Zum Beispiel lässt sich zeigen, dass bestimmte, populäre Angriffe Dienste betreffen, die als Teil des Grundschutzes automatisch vom ZID geschützt werden.

Bisherige Ergebnisse

Unser Honeynet zeichnet alle Pakete auf, die an eine der 4.096 IP-Adressen unseres Honeynets geschickt werden. Das erlaubt es uns, Statistiken zu erstellen, die zeigen, wie intensiv das Netzwerk der TU Wien Angriffen aus dem Internet ausgesetzt ist. Außerdem lässt sich verfolgen, welche Dienste zu einem bestimmten Zeitpunkt besonders interessant für Angreifer sind. Falls es zu einer plötzlichen Änderungen kommt und ein unbeachteter Dienst auf einmal besonders viel Netzwerkverkehr anzieht, kann das ein Hinweis darauf sein, dass eine neue Schwachstelle in diesem Dienst entdeckt worden ist, die von Angreifern aktiv ausgenützt wird. Der ZID könnte in diesem Fall reagieren und diesen Dienst zum Beispiel kurzfristig in den Grundschutz aufnehmen, um eine Infektion der Maschinen innerhalb der TU zu verhindern.

Die folgende Graphik zeigt die Gesamtanzahl der Verbindungen pro Tag für den Monat Oktober. Insgesamt wurden in den 31 Tagen 7.695.402 Verbindungen registriert (durchschnittlich fast 250.000 pro Tag). Sehr schön kann man die Schwankungen während der Woche erkennen; an Wochenenden wird typischerweise weniger Traffic gemessen.



In Tabelle 1 werden jene zehn Dienste (mit Portnummer) aufgelistet, die das häufigste Ziel von Verbindungen waren. Diese Tabelle zeigt, für welche Dienste sich Angreifer und Computerwürmer am meisten interessieren. Interessant ist, dass unter den Top-10 auch Port 135 aufscheint, der eigentlich durch den Grundschutz von außen nicht erreichbar ist. Ein Großteil der Verbindungen zu diesem Port kamen von Rechnern innerhalb des TU-Netzes, die durch die Firewall nicht blockiert werden.

Port	Verbindungen	Dienst
80	3.290.455	HTTP (Web)
5900	1.085.758	VNC Server (remote desktop)
22	292.511	Secure Shell (ssh)
135*	283.895	MS RPC
42	277.882	MS WINS (host name server)
3306	211.959	MySQL Database
4899	147.089	Remote Administrator (radmin)
8080	126.984	HTTP (Web)
5901	114.087	VNC Server Display:1
10000	97.843	Network Data Management Protocol

* Im Grundschutz gesperrter Port.

Tabelle 1

In Tabelle 2 werden jene Dienste aufgelistet, die am häufigsten in jenem Class C Netzwerk kontaktiert wurden, das ohne Grundschutz direkt aus dem Internet erreichbar ist. Es fällt auf, dass diese Liste Tabelle 1 sehr ähnlich ist. Der größte Unterschiede zeigt sich bei den Ports 139 und 4444. Beide Ports tauchen im Zusammenhang mit Wurmangriffen auf, 139 wird von Sasser angegriffen, 4444 wird vom Backdoor des Blaster Wurms verwendet.

Port	Verbindungen	Dienst
80	761.001	HTTP (Web)
139*	78.078	MS NetBIOS Session Service
5900	74.952	VNC Server (remote desktop)
135*	37.817	MS RPC
4444	34.066	Used by MS Blaster Worm
3306	14.490	MySQL Database
4899	8.705	Remote Administrator (radmin)
22	8.070	Secure Shell (ssh)
10000	8.014	Network Data Management Protocol
8080	7.715	HTTP (Web)

* Im Grundschutz gesperrter Port.

Tabelle 2

Seit das Honeynet in Betrieb genommen wurde, ist es uns gelungen, mittels nepenthes 180 unterschiedliche Malware Samples zu sammeln. Diese Samples, typischerweise Würmer, werden von uns im Rahmen unserer Forschung weiter analysiert. Außerdem leiten wir die Samples an Ikarus (einen Wiener Antivirushersteller) weiter, der dann bei Bedarf die Signaturdatenbank seines Virenscanners auf den neuesten Stand bringen kann.

Ein weiterer Nutzen des Honeynets liegt darin, dass wir jene Maschinen innerhalb des TU-Netzwerks identifizieren können, die das Netzwerk nach Diensten absuchen (scannen). Üblicherweise ist so ein Verhalten ein deutliches Zeichen dafür, dass die Maschine mit einem Computerwurm infiziert ist, der nach weiteren Opfern sucht. Wir übermitteln die Liste der gefundenen Scanner täglich und automatisch an die Security-Abteilung des ZID, der dann entsprechende Maßnahmen einleiten kann. Im Oktober haben wir insgesamt 24 infizierte Maschinen innerhalb des TU-Netzwerks entdeckt. Die Mehrzahl dieser Rechner waren interessanterweise Maschinen, die sich über das drahtlose Netzwerk oder über Einwahlleitungen von zu Hause mit dem TU-Netzwerk verbunden haben. Es waren aber auch einige Rechner von Universitätsinstituten darunter.

Zusammenfassung und Ausblick

Ein Honeynet kann nicht nur dabei helfen, die Trends von Angriffen im Internet zu beobachten und zu dokumentieren, sondern dient auch dazu, Malware zu sammeln, die sich erfolgreich im Netz verbreitet. Für unser Honeynet auf der TU Wien hat der ZID großzügigerweise einen Adressbereich von 4.096 IP-Adressen zur Verfügung gestellt. Unsere Honeypots registrieren im Durchschnitt täglich fast 250.000 Verbindungen und helfen mit, infizierte Maschinen im TU-Netz zu identifizieren.

Natürlich lässt sich das Honeynet noch weiter ausbauen. Zum Beispiel kann man die Anzahl der überwachten IP-Adressen oder die Menge jener Rechner, die nicht vom Grundschutz betroffen sind, erhöhen. Zusätzlich möchten wir das Honeynet noch mit Client-Honeypots erweitern. Anders als bei dem oben beschriebenen Netzwerk, wo Honeypots auf Angriffe von außen warten, surfen Client-Honeypots automatisiert im Internet. Das Ziel ist es, aktiv bösartige Seiten im Web zu entdecken, die Schwachstellen im Browser ausnützen. Wenn eine Menge von Client-Honeypots automatisch im Web surfen, kommt es natürlich zu einem hohen Paketaufkommen im Netzwerk. Auch hier sind wir auf die Unterstützung vom ZID angewiesen, der die entsprechende Bandbreite zur Verfügung stellen müsste.

References

[1] Lance Spitzner; Honeypots – Tracking Hackers; Addison-Wesley, 2003

[2] http://www.honeyd.org/

[3] http://nepenthes.mwcollect.org/

[4] http://honeytrap.sourceforge.net/

[5] http://www.vmware.com/

[6] http://www.seclab.tuwien.ac.at/

[7] http://www.zid.tuwien.ac.at/security/portsperren.php

IT Security, ein Praxisbericht

Ingmar Jaitner, Irmgard Husinsky[1]

Zur Aufrechterhaltung der IT-Sicherheit im TUNET und zur Unterstützung bei Schadensfällen existiert am ZID seit 1999 ein eigener Arbeitsbereich, der seit 2002 in die Abteilung Standardsoftware eingegliedert ist. Vordringlich muss der störungsfreie Betrieb innerhalb des TUNET sichergestellt sein. Dieser Artikel berichtet aus der täglichen Arbeit des Autors.

Einleitung

War in der Anfangszeit der Computer der Schutz vor einem Ausfall der Hardware wichtig, wurde dieser Aspekt durch die Verbesserung der Systeme in den Hintergrund gedrängt, dafür gab es vermehrt Störungen durch Softwarefehler sowie Sicherheitsprobleme im Softwarebereich aufgrund der höheren Komplexität gegenüber frühen Systemen.

Vor allem durch die Vernetzung der Systeme (Intranet und Internet) nahmen die Sicherheitsprobleme rasch zu. Bei Inselsystemen konnten Schadprogramme anfangs nur durch Datenaustausch, meist über Floppy Disks (z.B. Bootsektorviren), verbreitet werden. Mit Computern, die bis zu 24 Stunden pro Tag mit einem Netzwerk verbunden sind, steigt die Wahrscheinlichkeit, auf Sicherheitslücken untersucht und gegebenenfalls infiziert zu werden.

War es anfangs so, dass Einzelpersonen ausloten wollten, was alles möglich ist („ethical hackers"), teils aber auch Schaden anrichten wollten („crackers"), so rücken heute mehr und mehr kommerzielle Interessen in den Vordergrund. Dabei wird genau ausgeforscht, welche Infizierungspotentiale ein Rechner hat und welchen Aufwand es bedeutet, den Rechner unter Kontrolle zu bringen.

Schaden entsteht einerseits durch die Traffic-Belastung (Kosten für Bandbreite, Betriebsbeeinträchtigung), anderseits durch Datenverlust und den Aufwand zur Beseitigung von Infektionen (Datenmigration, Neuinstallation des Betriebssystems). Auch die Verschleierung der Identität wird versucht z. B. Botnetze[2], die einerseits kommerzielle Spammails oder Phishingmails (Mails, die die Gutgläubigkeit von Usern ausnutzen und Passwörter, PINs und TANs ausspionieren) versenden und DDOS (*Distributed Denial of Service*) Attacken (hunderte fernGesteuerte Botnetzrechner zusammen können ganze Webseiten lahm legen) durchführen.

Situation an der TU Wien

Die Security Policy der TU Wien regelt den Umgang mit Computern, die am Netz betrieben werden. Ferner sind die Betriebs- und Benutzungsordnung des ZID die TUNET-Benutzungsregelungen maßgeblich.
Siehe http://www.zid.tuwien.ac.at/security/policy.php

Für Anfragen und für Meldungen von Security-Problemen wurde gemäß RFC 2142 die E-Mail-Adresse security@tuwien.ac.at eingerichtet.

Die tägliche Arbeit im Security-Bereich am ZID umfasst:

- Beobachtung des Netzwerk-Traffics und der Firewall-Statistiken,
- Nachgehen von Hinweisen und Security Alerts,
- Zusammenarbeit mit den Systemadministratoren an den Instituten.

Während an einigen Instituten exzellentes Know-how vorhanden ist, benötigen andere Institute vermehrt Unterstützung durch den ZID. Hier gilt es vor allem, Sicherheitsbewusstsein zu bilden, in ständiger Zusammenarbeit mit den Systemadministratoren und Schulung derselben, vor allem bedingt durch die personelle Fluktuation.

Generell werden momentan Logfiles analysiert, auch die Systemadministratoren der Institute melden Vorkommnisse. Firewallstatistiken werden regelmäßig ausgewertet und helfen zur Entdeckung von Rechnern mit Securityproblemen. Auch die Geschwindigkeit der Entdeckung eines Sicherheitsvorfalls spielt eine Rolle.

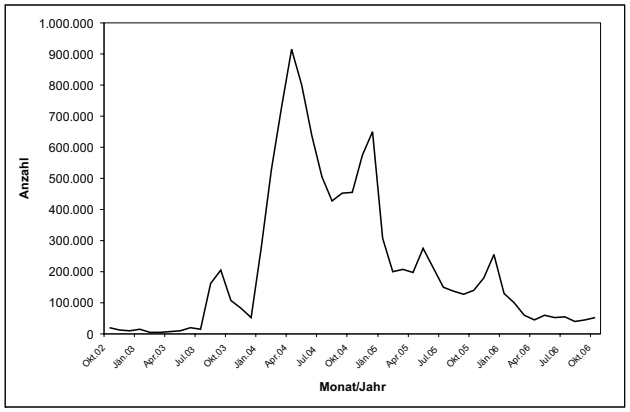
---

1 Redaktionelle Überarbeitung.

2 Botnet oder Bot-Netz (Kurzform von Roboter-Netzwerk): fernsteuerbares Netzwerk (im Internet) von PCs, welches aus untereinander kommunizierenden Bots (robot Programme) besteht. (Quelle: Wikipedia)

Im Folgenden sollen einige Arten von Vorkommnissen angeführt werden, die uns in letzter Zeit beschäftigten:

- Spam-Mail-Versand, Botnetze
  (Viren, Würmer und Trojaner):
  Hinweise erhält man über diverse Spam-Überwachungsorganisationen (z.B. Spamcop, jedoch ist die Trefferquote nur etwa 50%), weiters durch Analyse des Netzwerktraffics und von Logfiles.
  Bei Untersuchung des befallenen Rechners ergibt sich zumeist eine Infizierung des Rechners mit Viren, Würmern, Trojanern und Rootkits (Botnetzteilnehmer). Vor allem bei Multiinfektionen wird empfohlen, das Betriebssystem neu zu installieren, da bei Entfernung einzelner Viren ein Restrisiko bleibt, eine Lücke nicht geschlossen zu haben. Anschließend Kontrolle des Rechners durch ZID.
  Von Instituten werden Spam-Mails gemeldet, die explizit Viren enthalten, die nicht vom Spam-Filter markiert und bewertet wurden und wo Absenderadressen gesperrt wurden.



Anzahl der abgefangenen Viren am zentralen Virenscanner

- Rechner mit illegalen Diensten (die dem Administrator nicht bekannt sind, z.B. FTP-Server mit Raubkopien oder Filmen, Spieleserver, IRC Server):
  Solche Fälle werde durch Analyse des Netztraffics auffällig oder es langen Hinweise von außen ein.
  Verständigung des Administrators, Entfernung und Kontrolle ähnlich wie bei Spam-Mail (Botnetz) Fällen.

- Unangemeldete Rechner mit Securityproblemen (besonders unangenehm, da Verursacher schwer greifbar):
  Der Administrator des Subnetzes wird verständigt, ev. erfolgt eine Sperre. Damit die Services des TUNET in Anpruch genommen werden können, müssen alle Endgeräte angemeldet werden (TUNET Benutzerordnung, TUNET Datenbank).
  Die Ursachen sind hier versäumte Meldungen neu aufgesetzter Rechner, die zudem oft noch ohne Patches ins Netz gehen, sowie ganz bewusstes Herausgreifen von freien IP-Adressen von Usern, die z. B. ein (privates) Notebook anhängen wollen.

- DDOS Attacken von Botnetzteilnehmern (gehackten Rechnern) aus:
  Diese werden vom IDS (Intrusion Detection System) des Backbonebetreibers ACONet an uns gemeldet oder fallen über Netztraffic oder an Institutsfirewalls auf, sind allerdings dank der DDOS Prevention seitens ACONet seltener geworden, sowohl was die Angriffe als auch die an Botnetzattacken teilnehmenden Rechner betrifft.
  In den letzten 12 Monaten gab es keine aktiven Attacken.

- Auffälligkeiten bzgl. Netztraffic, Firewall, Logs ohne Störungen (auch gemeldete Angriffe von außen ohne Erfolg, aber z.B. mit Mitteilung an Betreiber):
  Typische Fälle sind hier Passwort Scanning, Portscanning und automatisierte Skripts, die über ganze IP-Adress-Ranges laufen.

- Phishing Attacken gewinnen an Bedeutung:
  Derzeit waren alle Fälle ohne gemeldeten finanziellen Schaden. Abhilfe bietet hier die Sperre der Absender und eine Spam-Markierungs-Gewichtung und die Aufmerksamkeit der Benutzer.

- Netcrawler Probleme (neu):
  verwässern die Firewall-Statistiken über die Anzahl der Connections pro 24 Stunden pro IP-Adresse. Da die Netcrawler Experimente vorwiegend von Informatikinstituten durchgeführt werden, haben wir mit den betroffenen Instituten vereinbart, dass Starts solcher Experimente an uns gemeldet werden.

- Securityprobleme mit Anwendungsprogrammen:
  Dies ist eine gefährliche und steigende Klasse von Securityproblemen, da sie oft unentdeckt bleiben.

- Aktives Cracking, von der TU ausgehend, „gewollt“ oder „probiert“ (war in den vergangenen Jahren stärker):
  Dies wird meist von Informatikstudenten ausgeführt, die dann sagen „Ich habe nicht gewusst, dass ich das nicht machen darf“. Nach Cracking aussehende Experimente.

- Copyrightfälle:
  Probleme gibt es bei unbeaufsichtigten Serverräumen und bei neuen Benutzern, die noch nicht mit der TUNET Policy vertraut sind. Entsprechende Maßnahmen werden gesetzt.

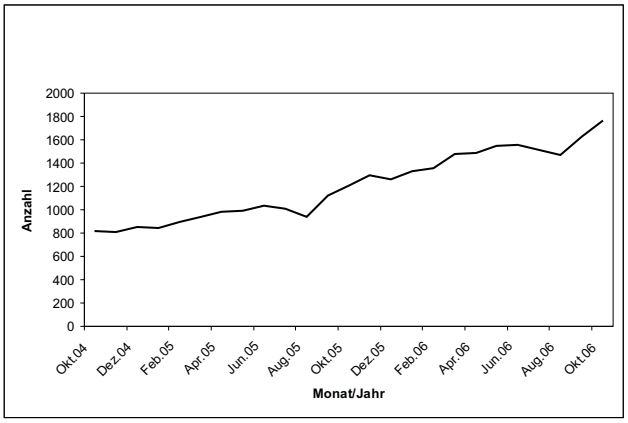
- Sonstige Anfragen und Probleme:
  Anfragen bzgl. Verstoß der Netiquette, TUNET Benutzerordnung und Security Policy, Rechtsprobleme, allgemeine Security Anfragen (z. B. Bitte um Sicherheitsscans) usw.

- Die Problematik von Skype Supernodes wird in einem eigenen Artikel behandelt (siehe Seite 9).

Wiederkehrende Ereignisse sind nicht nur der berühmte „Weihnachtshack“, auch Ende August/Anfang September gibt es eine Häufung von Securityereignissen, da über die Sommerferien viele Administratoren auf Urlaub sind und generell weniger Sicherheitsupdates eingespielt werden. Wenn zum Beispiel ein Client Mails von zwei bis drei Wochen abholt, ohne dass sichergestellt wurde, dass vorher Sicherheitsupdates installiert wurden, kann der Rechner infiziert werden.

Im Allgemeinen sind die Securityereignisse auf der TU Wien jedoch zurückgegangen. Dies ist vor allem auf Folgendes zurückzuführen:

- Großflächiger Einsatz von Firewalls an den Instituten.
- Regelmäßige Kontrolle auf den Firewalls.
- Die Systemadministratoren an den Instituten werden von der ZID Security laufend bei Fällen geschult (von angelernten Kräften bis absoluten Profis ist alles vorhanden) und haben ein hohes Maß an Problembewusstsein entwickelt (es erfolgt auch aktive Weitermeldung von verdächtigen Log-Einträgen).
- ACONET hat spezielle Detektierungstools, vor allem für DDOS und Botnetze und meldet verdächtige Rechner an uns weiter.



Anzahl der Rechner, die den WSUS Update Server in Anspruch genommen haben
Anmerkung: Hier sieht man trotz generell steigendem Aufkommen die Abschwächung in den Monaten Juli/August, was – wie im Artikel beschrieben – vermehrte Sicherheitsprobleme Ende August/Anfang September nach sich zieht

- Der ZID hat einen Windows Update Server installiert, wo Updates automatisch heruntergeladen, nach Prioritäten geordnet und auf Campuslizenzen verteilt und installiert werden, was unerfahrene User unterstützt und gestresste Administratoren entlastet.
- Nicht zuletzt haben auch die meisten Benutzer von Arbeitsplatz-Rechnern Sicherheitsproblembewusstsein entwickelt, da sie zum Teil auch schon unangenehme Erfahrungen beruflich oder privat machen mussten.

Von Sicherheitsproblemen betroffene Rechner werden rasch abgeschaltet, untersucht und in immer stärkerem Maß komplett neu aufgesetzt, da bei Multiinfektionen bei Entfernung mit Tools, wie es vor ein paar Jahren Standard war, die Wahrscheinlichkeit steigt, trotzdem eine Lücke nicht geschlossen zu haben und dann weiterer Schadcode nachgeladen und der Rechner wieder infiziert wird. Beim infizierten Rechner und nach Behebung wird ein Sicherheitsscan durchgeführt, um den Status zu prüfen. Die Kontrolle erfolgt netzwerkseitig von außen (Nessus) und von innen (z. B. mit alternativ bootbaren Betriebssystemen). Bei Wartungsverträgen erfolgt beides vom ZID, bei allen anderen nur die Kontrolle von außen vom ZID und die telefonische Beratung mit dem jeweiligen Administrator, der dann die Kontrolle von innen vornimmt.

Security wird immer komplexer. Nicht nur ist aus tausenden Security Alerts pro Jahr das Relevante und Wichtige herauszufiltern, insbesondere ist bei Universitäten das „innere“ und „äußere“ Netzwerk nicht sauber trennbar. In diesem Zusammenhang ist die Problematik der Notebooks und externen ADSL/VPN-Anschlüsse zu nennen, wo die Security wesentlich schwieriger als bei festen Institutsrechnern aufrechtzuerhalten ist, weil unerfahrene User die System administrieren und sich (bei Notebooks) in mehrere Netze einloggen. Auch PDAs und Handys werden zunehmend eingesetzt, wo Securityprobleme aber ebenfalls prinzipiell möglich sind (z. B. Java und offenes Bluetooth am Handy (ähnlich wie WLAN)), wo dann andere unter Umständen teure Telefongespräche führen können.

Die zuverlässige Erkennung von Securityereignissen wird mittelfristig eine große Rolle spielen, da zunehmend versucht wird, unauffälliger zu agieren. Maßnahmen wie ein Honeynet oder Intrusion Detection Systeme (IDS) sind zu prüfen. Vieles ist hier noch in Entwicklung. Gleichzeitig mit dem Einsatz solcher Maßnahmen sind der Datenschutz und die Verhältnismäßigkeit zu beachten.

Im Rahmen der vom ZID angebotenen Systempflege für Arbeitsplatzrechner an der TU (Wartungsvertrag) kann Unterstützung zur Prävention von Sicherheitsrisiken (und natürlich auch zur Schadensbehebung) angefordert werden. Weitere Informationen: http://sts.tuwien.ac.at/pss/.

Einrichtung eines Rechnerlabors für Virtuelle Produktentwicklung

Detlef Gerhard, Institut für Konstruktionswissenschaften und Technische Logistik

Derzeit wird am Institut für Konstruktionswissenschaften und Technische Logistik ein Rechnerlabor realisiert, das seit Beginn des Wintersemesters bereits Lehrveranstaltungen im Bereich CAD zur Verfügung steht, aber weiterhin ausgebaut wird, sodass ab dem kommenden Sommersemester mehrere CAx- und PDM-Softwarepakete für die Lehre zur Verfügung stehen werden. Der vorliegende Beitrag erläutert zusammenfassend die Hintergründe und Einzelheiten der Realisierung.

Ausgangssituation

Die Bedeutung der Informationstechnologie (IT), insbesondere der IT-Verfahren zur Unterstützung von Aufgaben und Tätigkeiten in den ingenieurwissenschaftlichen Prozessen industrieller Betriebe, hat in den letzten Jahren massiv zugenommen. Die Auswirkungen in Bezug auf Abläufe und Methoden zeigen sich in einer Neuorientierung in fast allen Bereichen eines Maschinenbauunternehmens. Klassische Arbeitsweisen werden geändert oder ersetzt und die Prozesse der Produktentstehung – innerbetrieblich oder, als Teil einer Supply Chain, unternehmensübergreifend mit Kooperationspartnern, Kunden und Lieferanten – werden neu definiert.

CAD-Systeme haben zwar schon vor einigen Jahren das manuelle Konstruieren am Zeichenbrett vollständig ersetzt, mittlerweile sind aber selbst die ursprünglich eingeführten 2D-CAD-Systeme zu einem großen Teil durch integrierte 3D-CAD-Systeme mit einem weit über die Geometriedefinition hinausgehenden Funktionsumfang ersetzt worden. Während der erste Schritt im Wesentlichen ein Übertragen alter Arbeitsweisen auf ein neues Hilfsmittel darstellte, ist die Umstellung von der 2D-CAD-Zeichnungserstellung auf die 3D-Produktmodellierung mit einer kompletten Änderung der Arbeitsweise für den Ingenieur verbunden.

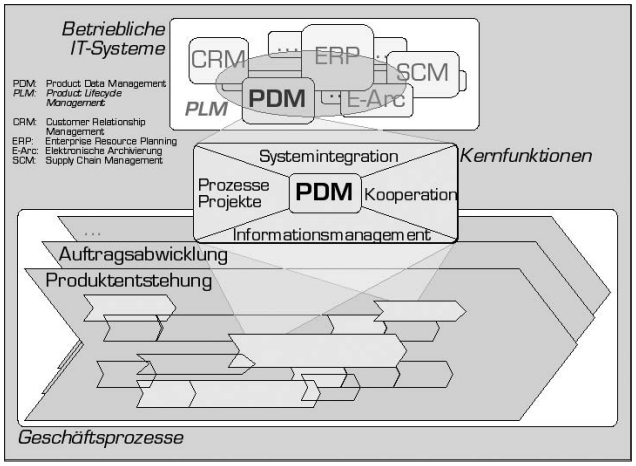
3D-Produktmodellierung eröffnet eine Vielzahl neuer Möglichkeiten: Bauteile können rechnerunterstützt analysiert (d. h. berechnet und beurteilt) und deren Verhalten simuliert werden. Untersuchungen mit Kinematik-Modulen für Kollisionsprüfungen beispielsweise im Rahmen von Ein- und Ausbauuntersuchungen sind möglich. Potenzielle Störgeometrien und andere Fehlerquellen können so frühzeitig ermittelt werden. Die mechanische Belastbarkeit von Bauteilen ebenso wie thermisches und strömungsmechanisches Verhalten kann mit Hilfe spezieller CAD-Anwendungsmodule auf Basis von Finite Elemente Methoden (FEM) nachgerechnet bzw. simuliert werden. Es können also bereits in einer frühen Entwicklungsphase Geometrien optimiert, Konstruktionen verifiziert und damit Fehler oder aufwändige Iterationen vermieden werden.

Im Rahmen der horizontalen Integration gesamter Prozessketten über die Entwicklung hinaus, wie beispielsweise im Bereich der Blechbiegeteile, lassen sich Abläufe von der Konstruktion über die automatisierte Erstellung von Abwicklungen in der Arbeitsvorbereitung bis hin zur Datenübergabe an entsprechende Fertigungsmaschinen durchgängig und ohne Medienbrüche abbilden. Damit werden Ergebnisse frühzeitig abgesichert und Durchlaufzeiten bzw. -kosten bei gleichzeitiger Qualitätssteigerung gesenkt.

Virtuelle Produktentwicklung

Der Begriff der „Virtuellen Produktentwicklung“ fasst die oben zum Ausdruck gebrachten Methoden inhaltlich zusammen. Dabei geht es im Kern um die möglichst vollständige Beschreibung eines realen Produkts mit all seinen Eigenschaften als Modell im Rechner und um das Management der erzeugten Daten sowie der Prozesse und verwendeten IT-Werkzeuge, damit eine möglichst lückenlose Dokumentation des Produkts und seines Entstehungsprozesses gewährleistet werden kann. Neben den CAx-Anwendungen nehmen in diesem Kontext die so

genannten Produktdatenmanagementsysteme (PDM) eine herausragende Stellung ein. Sie bilden die Integrationsplattform für die verschiedenen (daten-)erzeugenden Systeme, die unterschiedlichste Datenformate produzieren, bieten Funktionen für kooperative Arbeitsformen beispielsweise mit systemneutralen Viewern oder durch die Abbildung von rollenbasierten Zugriffsschutz-Mechanismen und bilden mit entsprechenden Workflow-Funktionen Prozesse wie Freigabe- oder Änderungswesen ab.



Einordnung und Funktion von PDM Systemen

Mit der Verfügbarkeit der Methoden der Virtuellen Produktentwicklung steigen selbstverständlich auch die Anforderungen an eine zukunftsorientierte Ausbildung von Studierenden ingenieurwissenschaftlicher Fächer. Aktuelle Studien belegen, dass bereits Anfang der 1980er Jahre Unternehmen ihren Mitarbeitern durchschnittlich eine Woche Schulung für den Umstieg auf das „elektronische Zeichenbrett“ 2D-CAD einräumten und dass dieser Wert bis heute ungefähr gleich geblieben ist. Das erforderliche Anwendungswissen, um die Möglichkeiten und Verfahren der Virtuellen Produktentwicklung effektiv in der betrieblichen Praxis einsetzen und nutzen zu können, ist jedoch um ein Vielfaches angestiegen.

Mit der Einrichtung des neuen Forschungsbereichs Maschinenbauinformatik und Virtuelle Produktentwicklung (MIVP) zum 01. 02. 2006 trägt die Fakultät für Maschinenwesen und Betriebswissenschaften im Bereich der Lehre diesen Punkten Rechnung. Auch in den neu eingerichteten Bakkalaureats- und Magisterstudienplänen sind entsprechende Veranstaltungen berücksichtigt worden. Die Zielsetzung ist, die klassische konstruktionswissenschaftliche Ausbildung, die das notwendige Fundament bildet, um das benötigte Methoden- und Anwendungswissen für die oben genannten Software-Applikationen und Spezialsysteme optimal im Sinne einer forschungsgeleiteten Lehre zu ergänzen. Obwohl am Institut für Konstruktionswissenschaften und Technische Logistik bereits seit Mitte der 1980er-Jahre erste Erfahrungen mit 3D-CAD Produkten gesammelt wurden und diese seit 1990 auch in der Lehre umgesetzt wurden, wird durch Schaffung des neuen Forschungsbereichs der Rechnerunterstützung ein neues Gewicht verliehen.

Die Basis für eine moderne Ausbildung bildet eine entsprechende Hard- und Software-Infrastruktur, auf die nachfolgend näher eingegangen wird.

Realisierung des Rechnerlabors

Bei der Planung des Rechner-Labors war entscheidend, dass die Arbeitsplatzrechner als Workstations mit Windows als Betriebssystem eingerichtet werden sollten. Der Grund hierfür ist insbesondere die Tatsache, dass ein Großteil der Anwendungssoftware entweder nur auf der Windows-Plattform zur Verfügung steht oder zumindest Windows die primäre Entwicklungsplattform der Systemhersteller ist, was eine langfristige Update-Fähigkeit und die Verfügbarkeit aktueller Versionen sicherstellt. Folgende Gruppen von Anwendungssoftware sollen im Labor für Virtuelle Produktentwicklung zum Einsatz kommen:

- Integrierte 3D-CAx-Systeme
- Spezialisierte Programme für Berechnung, Simulation und Visualisierung
- Produktdatenmanagementsysteme und andere betriebliche Informationssysteme
- Integrierte Software-Entwicklungsumgebungen (IDEs) für gängige Programmiersprachen wie Java oder C++

Um die bekannten Nachteile Windows-basierter Client/Server-Netzwerke im Hinblick auf die Sicherheitsproblematik und den Administrationsaufwand möglichst gering zu halten, wurden verschiedene Lösungen in Betracht gezogen und evaluiert. Dabei gab es auch Gespräche mit dem ZID über dessen Erfahrungen mit vorhandenen Systemen an der TU. Der Einsatz einer wartungsarmen Server-Based Computing Plattform, wie Windows Terminal Server oder Citrix Presentation Server, kam nicht in Betracht, da dieser Ansatz nicht für rechen- und insbesondere graphikintensive Applikationen, wie sie CAx-Systeme darstellen, geeignet ist. Das unter dem Codenamen „Tarpon“ angekündigte Application Streaming Produkt von Citrix, das Anwendungs-Software, analog zu der Verfahrensweise bei einem Video-Stream, vom Server lädt, ohne eine lokale Installation vornehmen zu müssen, war zum Zeitpunkt der Entscheidungsfindung noch nicht verfügbar, wird aber zukünftig sicher noch einmal separat evaluiert werden.

Konzept des Labors

Am Ende der Überlegungen fiel die Entscheidung, eine so genannte Software-Streaming Plattform der Fa. Ardence (www.ardence.com) einzusetzen, mit der schon positive Erfahrungen in dem EDV-Labor der Fakultät Bauingenieurwesen gemacht wurden.

Die Idee hinter der Ardence Desktop Edition ist im Wesentlichen, die Probleme bzw. die hohen Aufwendungen, die bei der Administration von PCs entstehen, dadurch zu vermeiden, dass sowohl das Betriebssystem als auch die Anwendungen über das Netzwerk zentralisiert bereitgestellt werden, anstelle über die lokale Festplatte

jedes einzelnen Rechners. Dadurch reduziert sich der Administrationsaufwand auf die Erstellung und Wartung (z. B. Neuinstallation, Patch-Management und Update von Softwareanwendungen) eines einzigen „Master"-Image, das beim Hochfahren der einzelnen Workstations vom Server geladen wird. Damit auf dem Server nur ein Arbeitsplatz-Image eingerichtet werden muss, müssen alle Arbeitsplatzrechner komplett gleich ausgestattet sein, was im vorliegenden Fall gegeben war. Es ist jedoch auch möglich, verschiedene Images für unterschiedliche Rechnergruppen, die nicht gleichartig ausgestattet sind, oder für unterschiedliche Einsatzzwecke mit entsprechenden Software-Konfigurationen, bereitzustellen. Ein weiterer Vorteil ist, dass alle Daten zwangsläufig auf dem Server und nicht lokal gespeichert werden müssen, was neben einer erhöhten Datensicherheit auch den Vorteil hat, das man nicht notwendigerweise immer am selben Arbeitsplatz arbeiten muss, um wirklich eine identische Arbeitsumgebung vorzufinden.

Versehentliche oder beabsichtigte Modifizierungen des Windows-Desktops sind nach jedem Rechnerneustart weg. Gefährdungen durch Computer-Viren oder Malware im Allgemeinen sind somit ebenfalls ausgeschaltet. Die eingebaute Festplatte kann deaktiviert werden, da sie für den Normalbetrieb nicht erforderlich ist, es sei denn, sie wird als lokaler Cache eingesetzt, um die Performance zu verbessern. Die Userdaten der Studenten werden auf den Studentenservern unter den jeweiligen Studentenaccounts gespeichert. Somit haben die Studenten alle ihre Daten zentral gespeichert und müssen sich nicht auf zusätzlichen Servern einloggen.

Das Rechnerlabor für Virtuelle Produktentwicklung

Hardware und Netzwerkinfrastruktur

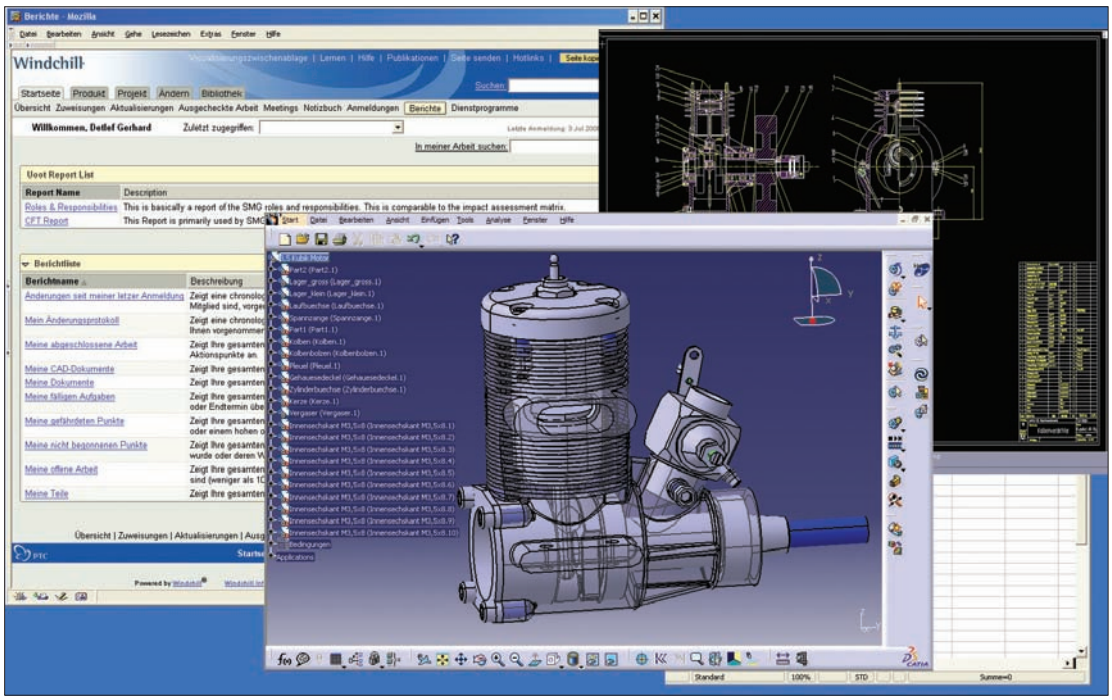
Die 18+1 Arbeitsplätze sind ausgestattet mit Intel-basierten Workstations der Firma HP (3,2 GHz Dual Core Prozessoren, 2GB RAM, NVIDIA FX1400 Graphikbeschleuniger) mit 20" Wide Screen TFT-Bildschirmen, Multimedia-Unterstützung. Zudem ist jeder Arbeitsplatz mit dem so genannten Space Traveller, einem spezialisierten 3D-CAD Eingabegerät der Fa. 3DConnexion (www.3dconnexion.de) für Pan, Zoom und Rotate Funktion, ausgestattet, mit dem 3D-Modelle beliebig im Raum orientiert werden können, ohne CAD-Funktionen ab- oder unterbrechen zu müssen.

Die Leistungsfähigkeit im Netzwerk wird durch eine durchgängige GBit-Verbindung zwischen Desktop-Workstations und Server gewährleistet. Die Serverseite ist entsprechend leistungsfähig ausgelegt, da Server im verwendeten Konzept ein „single point of failure" darstellen. Neben der Funktion des Ardence Servers werden für das Rechner-Labor für die PDM-Systemumgebung Datenbank- und Applikationsserver benötigt. Für die genannten Aufgaben werden Doppelprozessormaschinen der Firma HP mit bis zu 8 GB RAM und Windows 2003 Server als Betriebssystem eingesetzt. Eine externe Festplatteneinheit sorgt für ca. ein Terabyte Speicherkapazität der Anlage. Eine entsprechende Daten- bzw. Ausfallsicherheit wird durch getrennte RAID-Systeme, jeweils für Betriebssystem und Datenbereich, erreicht. Alle Server sind klimatisiert untergebracht und durch redundant ausgelegte Netzteile und Kühler sowie durch eine unterbrechungsfreie Stromversorgung abgesichert.

Zur Benutzer-Authentifizierung über den TU-Account der Studierenden wird das Programm PGina verwendet (http://sourceforge.net/projects/pgina). PGina ersetzt die standardmäßig auf der Windows-Plattform verwendete GINA (Graphical Identification aNd Authentication) DLL durch einen entsprechenden Ersatz, mit dem über so genannte Plugins auf unterschiedliche Weise eine Authentifizierung durchgeführt kann, beispielsweise in dem UNIX-Server oder LDAP Directories angesprochen werden. Auf diese Weise können alle Studierenden mit gültigem Studentenaccount validiert werden, ohne dass eine separate Benutzerverwaltung aufgebaut werden muss. Dies ist ein entscheidender Vorteil in Bezug auf den Administrationsaufwand. Eine Überprüfung der Accounts im Hinblick auf die Teilnahme an bestimmten Veranstaltungen und damit verbundene Rechte ist derzeit über dieses System noch nicht möglich aber auch nicht notwendig, da der Zugang zum Rechner-Labor limitiert ist.

Anwendungs-Software

Ein Grundsatz im Bereich der Lehre ist, dass keine „Spezialisten" für eine bestimmte Anwendungssoftware ausgebildet werden sollen, sondern dass den Studierenden ein in erster Linie systemneutraler bzw. systemübergreifender Zugang zu den Applikationen vermittelt werden soll. Jedes CAx-System hat seine Stärken und Schwächen und in Bezug auf die zu verwendende Methodik ergeben sich enorme Unterschiede in Abhängigkeit der Art, wie bestimmte Systeme funktionieren. Wichtig ist, ein entsprechendes Methodenwissen zu vermitteln und hier liegt der Schwerpunkt im Rahmen der Ausbildung. Je nach Projekt, in das Studierende an den verschiedenen Instituten der Fakultät eingebunden sind, sind ohnehin bestimmte Systeme vorgegeben. Im Rechnerlabor sind verschiedene, marktführende Systeme verfügbar, teilweise durch Nutzung von Campus-Lizenzen, teilweise durch individuell mit den Systemherstellern verhandelte Abkommen. Im Bereich der integrierten 3D-CAD/CAE/CAM-Systeme sind dies insbesondere:



Beispiel Desktop

- Catia (IBM/Dassault)
- One Space Designer (CoCreate)
- Pro/ENGINEER Wildfire (PTC)
- Solid Edge (Unigraphics)

Der PDM-Bereich wird zunächst mit den Systemen Windchill der Fa. PTC (www.ptc.com) und CIM DATABASE von Contact Software (www.contact.de) umgesetzt. Windchill nutzt konsequent Web-Technologien und besteht aus den Hauptkomponenten PDMLink für das Management produktbezogener Daten und die Integration von CAx-Systemen und ProjectLink, das als Plattform für teamorientiertes Arbeiten dient. CIM DATABASE ist eine plattform- und CAD-unabhängige PDM-Lösung zur Verwaltung aller technischen Dokumente, Daten, Abläufe und Projekte in Unternehmen. Sie zeichnet sich ebenfalls durch eine moderne, skalierbare Systemtechnologie aus, verfügt über Standardintegrationen zu zahlreichen CAD- und ERP-Systemen und wird komplett aus Deutschland (Bremen) heraus entwickelt und vertrieben.

Mit PTC konnte im Rahmen der Einrichtung des Rechnerlabors eine Campus-Lizenz (500 User) verhandelt werden, die Anwendern an der gesamten TU zur Verfügung steht. Im Rechnerlabor wird eine entsprechende Windchill-Standardumgebung implementiert, die bei Interesse auch anderen Instituten zur Nutzung zur Verfügung gestellt werden kann. Beide Systeme liegen bzgl. der Kosten für die zur Verfügung stehenden Lizenzen normalerweise, d. h. außerhalb des Hochschulbereichs, weit im sechsstelligen Euro-Bereich.

Zusammenfassung

Der Betrieb des Rechner-Labors konnte nach Entscheidung des Umsetzungskonzepts im Juni für die CAD-Ausbildung rechtzeitig zum Wintersemester 2006 in Betrieb genommen werden und läuft seitdem problemlos. Für die Implementierung der Lösung wurde auf die Erfahrungen zurückgegriffen, die bereits beim Einrichten eines EDV-Labors im Bereich Bauingenieurwesen gemacht wurden. Für die Unterstützung, auch seitens des ZID, sei an dieser Stelle herzlich gedankt. Die wesentlichen Vorteile der gewählten Software-Streaming Plattform sind einfache Softwareinstallation an den Arbeitsplatzrechnern, das Synchronhalten aller Rechner mit ihrer installierten Software und eine Konzentration auf die Serverwartung.

Die leistungsfähigen Software-Produkte, die guten Kontakte zu Systemanbietern und die Beschäftigung mit der hochaktuellen Thematik PDM ermöglichen eine moderne Ausbildung mit allen Methoden und Werkzeugen, die heute in der Industrie im Rahmen des Produktentwicklungsprozesses eingesetzt werden. Dies bildet auch eine gute Basis für zukunftsweisende Projekt-/Diplomarbeiten und Forschungsprojekte. Da die Entwicklung heutiger „mechatronischer“ Produkte immer eine Integration von mechanischer Konstruktion (M-CAD) mit elektrotechnischer Konstruktion (E-CAD) sowie der Entwicklung von Software erfordert, wäre es wünschenswert, wenn im Rahmen einer interfakultären Zusammenarbeit dieser ganzheitliche Ansatz auch verstärkt in die Lehre einfließen könnte. Möglicherweise bietet die gemeinsame Nutzungsmöglichkeit eines PDM-Systems auch hier eine Integrationsplattform als Basis für derartige Aktivitäten.

Kontakt:

Institut für Konstruktionswissenschaften
und Technische Logistik
Forschungsbereich Maschinenbauinformatik
und Virtuelle Produktentwicklung
Univ.-Prof. Dr.-Ing. Detlef Gerhard
http://www.mivp.tuwien.ac.at/

Personalnachrichten

Herr **Siegfried Loibner** arbeitet seit Anfang Juni als Vertetung für Frau Natalie Kamenik, die sich zurzeit in Karenz befindet, im Bereich Campus Software Setup.

Wir wünschen ihm viel Erfolg und Freude bei seiner Tätigkeit am ZID.

Zur Betreuung der Internet-Räume und im Service Center sind am ZID folgende Studienassistenten angestellt:

C. Fellinger
J. Greilberger
M. Hofer
M. Jaros
H. Judt
P. Kotik
P. Lischka
T. Wojcik
Dipl.-Ing. M. Wögerbauer
K. Wong

Herr **Walter Haider** hat am 1. 12. 2006 die Freizeitphase seiner Altersteilzeitkarenz angetreten.

Nach dem Abschluss seines Studiums der Rechentechnik im Jahr 1973 begann seine berufliche Laufbahn am Rechenzentrum der TU Wien, Abt. Digitalrechenanlage. Herr Haider war vor allem für die Betreuung von Softwarepaketen (Installation, Anpassungen an die verschiedenen Betriebssysteme, Dokumentationen) zuständig. Seit 1990 arbeitete Herr Haider am ZID zunächst in der Abteilung Institutsunterstützung, dann in der Abteilung Zentrale Services und betreute zusätzlich zu seinen Software-Aufgaben die SGI-Systeme als Systemadministrator.

Wir möchten uns bei Walter Haider für seinen großen Einsatz und sein Engagement bei der Lösung von technischen und sozialen Problemen herzlich bedanken und wünschen ihm für diesen neuen Lebensabschnitt vor allem Gesundheit und alles Gute.

Auskünfte, Störungsmeldungen: Service Center

Bitte wenden Sie sich bei allen Fragen und Problemen, die das Service-Angebot des ZID betreffen, zunächst an das Service Center.

Telefon: 58801- **42002**

Adresse: 1040 Wien, Wiedner Hauptstraße 8-10, Freihaus, 2.OG, gelber Bereich
Montag bis Freitag, 8 bis 17 Uhr

Ticket System: **https://service.zid.tuwien.ac.at/support/**

E-Mail-Adressen:
für Auskünfte und Störungsmeldungen

office@zid.tuwien.ac.at	allgemeine Anfragen
trouble@noc.tuwien.ac.at	TUNET Störungen
hostmaster@noc.tuwien.ac.at	TUNET Rechneranmeldung / Administrativa
telekom@noc.tuwien.ac.at	Telefonie
adslhelp@zid.tuwien.ac.at	TU-ADSL Beratung
security@tuwien.ac.at	Netz- und Systemsicherheit
pss@zid.tuwien.ac.at	Systemunterstützung
operator@zid.tuwien.ac.at	Operating zentrale Server
mailhelp@zid.tuwien.ac.at	Mailbox-Service
studhelp@zid.tuwien.ac.at	Internet-Räume
tuwis@zv.tuwien.ac.at	TUWIS++

Telefonliste, E-Mail-Adressen

Zentraler Informatikdienst (ZID)
der Technischen Universität Wien
Wiedner Hauptstraße 8-10 / E020, 1040 Wien
Tel.: (01) 58801-42002
Fax: (01) 58801-42099
Web: www.zid.tuwien.ac.at

Leiter des Zentralen Informatikdienstes:

W. Kleinert	42010	kleinert@zid.tuwien.ac.at

Administration:

S. Freisleben	42015	freisleben@zid.tuwien.ac.at
A. Müller	42015	mueller@zid.tuwien.ac.at
M. Grebhann-Haas	42018	grebhann-haas@zid.tuwien.ac.at

Öffentlichkeitsarbeit

I. Husinsky	42014	husinsky@zid.tuwien.ac.at
I. Macsek	42047	macsek@zid.tuwien.ac.at

Service Center

Leitung:

Ph. Kolmann	42011	kolmann@zid.tuwien.ac.at
Th. Pitlik	42012	pitlik@zid.tuwien.ac.at
H. Ehrhardt	42066	ehrhardt@zid.tuwien.ac.at
S. Geringer	42065	geringer@zid.tuwien.ac.at
M. Markowitsch	42062	markowitsch@zid.tuwien.ac.at
S. Bachinger		bachinger@zid.tuwien.ac.at
P. Eisele		eisele@zid.tuwien.ac.at
D. Österreicher		oesterreicher@zid.tuwien.ac.at
K. Pegac		pegac@zid.tuwien.ac.at
D. Sabounji		sabounji@zid.tuwien.ac.at
A. Sorger		sorger@zid.tuwien.ac.at

ADV-Abteilung

www.zid.tuwien.ac.at/adv/

Leitung:

E. Dvorak	41070	dvorak@zid.tuwien.ac.at
M. Beer	41077	mbeer@zid.tuwien.ac.at
B. Borovali	41072	borovali@zid.tuwien.ac.at
J. Divisch	41079	divisch@zid.tuwien.ac.at
F. Glaser	41074	glaser@zid.tuwien.ac.at
S. Gründlinger	41194	gruendlinger@zid.tuwien.ac.at
A. Knarek	41075	knarek@zid.tuwien.ac.at
D. Lyzczarz	41076	lyzczarz@zid.tuwien.ac.at
W. Niedermayer	41195	niedermayer@zid.tuwien.ac.at
A. Rajkovats	41073	rajkovats@zid.tuwien.ac.at
R. Vargason	41196	vargason@zid.tuwien.ac.at
M. Wograndl	41078	wograndl@zid.tuwien.ac.at

Abteilung Standardsoftware

sts.tuwien.ac.at

Leitung

A. Blauensteiner	42020	blauensteiner@zid.tuwien.ac.at
Ch. Beisteiner	42021	beisteiner@zid.tuwien.ac.at
J. Donatowicz	42028	donatowicz@zid.tuwien.ac.at
G. Gollmann	42022	gollmann@zid.tuwien.ac.at
M. Holzinger	42025	holzinger@zid.tuwien.ac.at
I. Jaitner	42037	jaitner@zid.tuwien.ac.at
A. Klauda	42024	klauda@zid.tuwien.ac.at
R. Ladner	42033	ladner@zid.tuwien.ac.at
S. Loibner	42034	loibner@zid.tuwien.ac.at
H. Mastal	42079	mastal@zid.tuwien.ac.at
H. Mayer	42027	mayer@zid.tuwien.ac.at
Th. Mikulka	42023	mikulka@zid.tuwien.ac.at
E. Schörg	42029	schoerg@zid.tuwien.ac.at
R. Sedlaczek	42030	sedlaczek@zid.tuwien.ac.at
W. Selos	42031	selos@zid.tuwien.ac.at
B. Simon	42032	simon@zid.tuwien.ac.at
W. Steinmann	42036	steinmann@zid.tuwien.ac.at
P. Torzicky	42035	torzicky@zid.tuwien.ac.at

Abteilung Kommunikation

nic.tuwien.ac.at

Leitung

J. Demel	42040	demel@zid.tuwien.ac.at
F. Blöser	42041	bloeser@zid.tuwien.ac.at
G. Bruckner	42046	bruckner@zid.tuwien.ac.at
Th. Eigner	42052	eigner@zid.tuwien.ac.at
Th. Gonschorowski	42056	gonschorowski@zid.tuwien.ac.at
J. Haider	42043	jhaider@zid.tuwien.ac.at
P. Hasler	42044	hasler@zid.tuwien.ac.at
G. Kittel	42042	kittel@zid.tuwien.ac.at
J. Kainrath	42045	kainrath@zid.tuwien.ac.at
J. Klasek	42049	klasek@zid.tuwien.ac.at
W. Koch	42053	koch@zid.tuwien.ac.at
F. Matasovic	42048	matasovic@zid.tuwien.ac.at
W. Meyer	42050	meyer@zid.tuwien.ac.at
J. Öttl	42057	oettl@zid.tuwien.ac.at
Ch. Schwarz	42055	schwarz@zid.tuwien.ac.at
A. Straschil	42057	straschil@zid.tuwien.ac.at
R. Vojta	42054	vojta@zid.tuwien.ac.at
Michael Weiss	42058	mweiss@zid.tuwien.ac.at
Walter Weiss	42051	weiss@zid.tuwien.ac.at

Abteilung Zentrale Services

www.zserv.tuwien.ac.at

Leitung

P. Berger	42070	berger@zid.tuwien.ac.at
W. Altfahrt	42072	altfahrt@zid.tuwien.ac.at
J. Beiglböck	42071	beiglboeck@zid.tuwien.ac.at
P. Deinlein	42074	deinlein@zid.tuwien.ac.at
P. Egler	42094	egler@zid.tuwien.ac.at
C. Felber	42083	felber@zid.tuwien.ac.at
H. Flamm	42092	flamm@zid.tuwien.ac.at
E. Haunschmid	42080	haunschmid@zid.tuwien.ac.at
M. Hofbauer	42085	hofbauer@zid.tuwien.ac.at
F. Mayer	42082	fmayer@zid.tuwien.ac.at
J. Pfennig	42076	pfennig@zid.tuwien.ac.at
M. Rathmayer	42086	rathmayer@zid.tuwien.ac.at
M. Roth	42091	roth@zid.tuwien.ac.at
J. Sadovsky	42073	sadovsky@zid.tuwien.ac.at
D. Sonnleitner	42087	sonnleitner@zid.tuwien.ac.at
Werner Weiss	42077	weisswer@zid.tuwien.ac.at

Service Center

Wo: Zentraler Informatikdienst
1040 Wien, Wiedner Hauptstraße 8-10
Freihaus, 2.OG, gelber Bereich

Wann: Montag bis Freitag, 8 bis 17 Uhr
Telefon: 58801 - 42002

Auskünfte, Störungsmeldungen

Bitte wenden Sie sich bei allen Fragen und Problemen, die das Service-Angebot des ZID betreffen, zunächst an das Service Center.

Anfragen im Web (Ticket System):

https://service.zid.tuwien.ac.at/support/