Beitragsüberschrift

Wie zuverlässig ist der Empfang von Netzwerkpaketen bei der Verwendung des Transportprotokolls UDP? *

* Der Beitrag basiert auf einer Masterarbeit an der Fachhochschule Stralsund in Kooperation mit dem Max-Planck-Institut für Plasmaphysik [1]

Autoren

Steffen Tambach

Steffen.Tambach@tambach.net

Steffen Tambach hat seine Arbeit zum Master of Science an der Fachhochschule Stralsund zum Thema „Analyse eines verbindungslosen Netzwerkprotokolls der Gruppenkommunikation in Ethernet-basierten Netzwerken“ geschrieben.

Christine Hennig

Max-Planck-Institut für Plasmaphysik

Teilinstitut Greifswald

Wendelsteinstraße 1

17491 Greifswald

Tel. 03834 88 2410

Fax. 03834 88 2509

Christine.Hennig@ipp.mpg.de

Christine Hennig arbeitet im Projekt XDV (Experimentnahe Datenverarbeitung).

Korrekturadresse

Christine Hennig

Max-Planck-Institut für Plasmaphysik

Teilinstitut Greifswald

Wendelsteinstraße 1

17491 Greifswald

Tel. +49 3834 88 2410

Fax +49 3834 88 2509

E-Mail: Christine.Hennig@ipp.mpg.de

Kurztitel

Wie zuverlässig ist UDP? How reliable is UDP?

English short abstract

The paper introduces a multicast application for continuous data acquisition: a distributed monitoring system to observe a large physical experiment. It has high demands on real time capability as well as scalability to the number of receivers.

The implementation is based on existing UDP and IP Multicast internet technology. UDP data losses had been observed at a receiving Personal Computer leading to the question: How reliable is UDP? The losses are examined and ways to overcome the losses are shown exemplarily on a Linux operating system.

Abstract

Im vorliegenden Beitrag wird eine Multicast-Anwendung für eine kontinuierliche Datenerfassung vorgestellt. Es handelt sich um ein verteiltes Monitoring-System zur Beobachtung eines physikalischen Großexperiments. Dieses System stellt u.a. Echtzeit-Anforderungen an die Datenübertragung und soll auf endliche Empfänger skalieren. Eine Schlüsselrolle spielen hierbei der Empfang und die Weiterverarbeitung der Daten.

Lassen sich derartige Anforderungen mit der Verwendung von UDP-basierter Internet-Technologie realisieren? Im konkreten Fall wurde eine verbindungslose Kommunikation auf der Basis von UDP und IP-Multicast implementiert. Dabei wurden Datenverluste festgestellt, die im vorliegenden Artikel untersucht werden und zu der verallgemeinerten Frage führen: Wie zuverlässig ist die Verwendung - insbesondere die Empfangsverlustrate - von UDP-basierter Gruppenkommunikation?

Am Beispiel des Betriebssystems Linux wird nachvollzogen, wie der Empfang und die Verarbeitung von UDP-Datenpaketen auf einem i386-basierten Rechnersystem erfolgen, wann Datenverluste auftreten können und wie sich diese vermeiden lassen.

Schlüsselwörter

UDP, IP-Multicast, Linux, BSD-Socket, Monitoring, Kontinuierliche Datenerfassung

1. Einführung

Am Max-Planck-Institut für Plasmaphysik ist ein Großexperiment im Aufbau. Dessen Experimentablauf soll über ein verteiltes Monitoring-System dargestellt werden. Die hierzu notwendigen, physikalisch relevanten Informationen stammen aus verteilten Messquellen und sollen einer Gruppe von Beobachtern zeitnah zur visuellen Inspektion an verschiedenen Orten über das lokale Netzwerk zur Verfügung stehen.

Hierfür wird ein Gruppenkommunikations-System eingesetzt, wie es ähnlich auch bei Videokonferenzen zwischen mehreren Partnern, VoIP (engl. Voice over IP) oder ereignisgesteuerten Architekturen [2] verwendet wird. Solchen Systemen ist gemeinsam, dass die Übertragungen keine spürbaren Verzögerungen haben dürfen. Ebenso wenig lassen sich Verluste tolerieren, die eine Verzerrung oder Sinnentstellung der Inhalte zur Folge haben, wie beispielsweise verschluckte Silben oder das Fehlen bzw. die Verfälschung relevanter Bildinformationen.

Bei einer Punkt-zu-Punkt-basierten Kommunikation wird allgemein TCP/IP angewandt. Das Transportprotokoll TCP (engl. Transmission Control Protocol) mit seinen Kontrollfluss-Mechanismen sorgt u.a. dafür, dass alle Datenpakete den Empfänger erreichen. Jedoch ist TCP/IP in speziellen Anwendungsfällen nicht das Mittel der Wahl, d.h. TCP genügt insbesondere nicht einer Echtzeitanforderung. Eine alternative und einfache Realisierung besteht darin, dass die Quelle die Datenpakete an potentielle Empfänger sendet, ohne sich um den korrekten Empfang zu kümmern. Die Empfänger sammeln die ankommenden Datenpakete ein. Dieses Prinzip wird mittels des verbindungslosen Transportprotokolls UDP (engl. User Datagram Protocol) realisiert und kommt bei den o.g. Anwendungen zum Einsatz. Eventuelles erneutes Senden einer veralteten Information zu Lasten aktueller Informationen wird damit vermieden, Datenverluste werden hingegen nicht behandelt. Unter diesem Aspekt steht auch die IHK-Prüfungsfrage an Fachinformatik-Auszubildende der Fachrichtung Systemintegration (vgl. Abschlussprüfung Sommer 2006): „Erläutern Sie, warum UDP als Transportprotokoll für VoIP besonders geeignet ist“. [3]

Im Falle mehrerer Empfänger ist es darüber hinaus sinnvoll, identische Informationen auf demselben Weg nicht mehrfach zu übertragen. In diesem Anwendungsfall muss Multicast anstelle von Unicast verwendet werden. Eine IP-Multicast-Adresse fungiert als virtuelle IP-Adresse der Empfänger-Gruppe, der sich die einzelnen Empfänger anschließen. Der Sender sendet nur einmal an die (Multicast) IP-Adresse.

UDP garantiert nicht die Zuverlässigkeit einer Datenübertragung, d.h. Datenverluste sind prinzipiell nicht auszuschließen. Obwohl bei VoIP mit UDP keine Datenverluste beschrieben sind, gibt es sie bei höheren Datenraten wie für das Monitoring-System. In diesem Artikel wird dargelegt, wann und wo UDP-Datenverluste auftreten und wie sie minimiert werden können.

2. Das Max-Planck-Institut für Plasmaphysik und das Experiment Wendelstein 7-X

Das Max-Planck-Institut für Plasmaphysik an den Standorten Garching und Greifswald erforscht die Grundlagen für ein Fusionskraftwerk nach dem Prinzip des magnetischen Einschlusses. Ähnlich wie unsere Sonne soll ein zukünftiges Fusionskraftwerk Energie aus der Verschmelzung von leichten Atomkernen gewinnen (vgl. Abbildung 1).

Abbildung 1: Die Fusionsreaktion

Wegen seiner extrem hohen Temperatur kann ein Fusionsplasma nicht unmittelbar in materiellen Gefäßen eingeschlossen werden. Stattdessen wird ein geeignet geformter Magnetfeldkäfig verwendet. Am Teilinstitut Greifswald ist das Fusionsexperiment *Wendelstein 7-X* in Bau. Abbildung 2 zeigt eine dreidimensionale Darstellung der Geometrie von Plasma und Spulen.

Abbildung 2: Magnetspulen und Plasma von Wendelstein 7-X

Die extremen Bedingungen in einem Fusionsplasma erfordern besondere Messmethoden, um seinen Zustand – Temperatur, Dichte, Energieinhalt, Ströme im Plasma, Verunreinigungen usw. – zu untersuchen. Dazu werden speziell entwickelte Diagnostik-Verfahren eingesetzt. Das bedeutet den Einsatz mehrerer alternativer Verfahren zur Messung derselben physikalischen Größe, um systembedingte Messfehler zu minimieren. [4]

3. Experimentnahe Datenverarbeitung

Im Gegensatz zu den Fusions-Experimenten, die im sog. Schussbetrieb gepulst jeweils wenige Sekunden bis Minuten arbeiten, wird der Wendelstein 7-X kontinuierlich über etwa 30 Minuten betrieben werden und erfordert die Entwicklung neuartiger Datenerfassungstechniken: Bei den schussbetriebenen Experimenten werden die digitalisierten Daten für den Schusszeitraum in entsprechenden lokalen Speichern der Messgeräte akkumuliert und nach Beendigung des Schusses ausgelesen und verarbeitet. Hingegen ist es beim kontinuierlichen Experimentierbetrieb wie dem des Wendelstein 7-X unmöglich, Daten über den gesamten Zeitraum lokal zwischen zu speichern. Die Daten müssen kontinuierlich ausgelesen und die lokalen Caches zyklisch geleert werden.

Das erwartete Datenvolumen aller Sensoren der verschiedenen Diagnostiken beträgt nach der Digitalisierung und geeigneter Datenreduktion mehrere TBytes innerhalb eines halbstündigen Experimentlaufs. Die digitalisierten Daten sind zeitnah und vollständig aufzunehmen, zu

verarbeiten, weiterzuleiten und zu speichern. Ein Teil dieser Daten wird zur Überwachung und Steuerung des weiteren Experimentverlaufs sofort benötigt. Alle Daten bzw. Informationen müssen archiviert werden und für spätere tiefgründige Analysen zur Verfügung stehen.

4. Monitoring

Wesentlicher Bestandteil der Experimentnahen Datenverarbeitung ist das integrierte Monitoring-System. Es ermöglicht, einen signifikanten Teil der Daten zeitnah visualisieren zu können und damit die wesentlichen physikalischen Phänomene zu beobachten und den momentanen Experimentverlauf darzustellen. Dazu müssen repräsentative Daten erzeugt und dargestellt werden. Zur Laufzeit des Experiments werden dafür die für das Monitoring geeigneten und gewünschten Daten ausgewählt, die Informationen ausgedünnt, verarbeitet und für das menschliche Auge entsprechend visualisiert. [5]

Abbildung 3: Schematische Darstellung des Datenflusses beim Monitoring

Die Monitoring-Daten sollen innerhalb des lokalen Netzwerkes experimentsynchron an beliebig viele Teilnehmer verteilt werden. Die dafür ausgewählten Datenerfassungen schicken einen ausgedünnten Monitor-Datenstrom über das Netzwerk an einen oder mehrere Monitor-Server. Diese verarbeiten die Daten, kombinieren sie nach Bedarf mit weiteren Daten aus anderen Quellen und verschicken die aufbereiteten Daten wiederum über das Netzwerk an die Clienten. Die Monitor-Clienten ihrerseits lesen die ausgewählten Daten vom Netzwerk und stellen diese in geeigneter Form dar (vgl. Abbildung 3).

Da die Monitor-Daten von mehreren Datenerfassungs-Stationen zu den Monitor-Servern und insbesondere vom jeweiligen Monitor-Server zu mehreren Monitor-Clienten unterwegs sind, handelt es sich hierbei um eine Gruppenkommunikation. Eine zeitnahe Verfügbarkeit hat Vorrang vor der Vollständigkeit der Datenübertragung. Der Datentransport im lokalen Netzwerk wurde über den UDP/IP-Multicast Übertragungsmechanismus realisiert, wodurch die Vervielfachung des Datentransportes über identische Strecken vermieden wird.

Sowohl die qualitativen Anforderungen als auch die technischen Realisierungsmöglichkeiten entsprechen also denen der eingangs erwähnten VoIP-Technik.

5. Kommunikations-Datenverluste beim Monitoring

Bei Performance-Tests des Monitoring-Systems zeigte sich, dass bei hohen Datenraten am Monitor-Clients Datenverluste auftraten. Diese Verluste waren nicht punktuell, sondern betrafen komplette Zeitbereiche und können zu einer Verfälschung von Signalen führen, wie die Abbildung 4 am Beispiel eines Sinus-Signals illustriert.

Abbildung 4: Darstellung eines Sinus-Signals in der Monitoring-Darstellung ohne Datenverlust (links) und mit Datenverlust (rechts)

Bei Betrachtung der verfügbaren Bandbreite waren diese Verluste völlig unerwartet. Die Datentransferraten bei den Tests betrugen circa 20 Mbit/s bei einer maximal möglichen Bandbreite aller Netzwerkkomponenten von 100 Mbit/s und einem sternförmigen geschwitchten Backbone mit 1 Gbit/s Bandbreite. Dennoch sind die Datenraten, die beim Experiment-Monitoring anfallen, verglichen mit den bei VoIP anfallenden Daten, die sich im Bereich zweistelliger Kbit/s bewegen, als vergleichsweise hoch einzustufen.

6. Untersuchung der UDP-Datenverluste

6.1. Die Testprogramme und Testumgebungen

Um die Verlustursache(n) zu verifizieren, ist es in einem ersten Schritt erforderlich, alle potentiellen Fehlerquellen separat zu untersuchen. Dazu wurde eine anwendungsbezogene Multicast-Testprogrammsuite erstellt. Diese bestand u.a. aus einem Senderprogramm, das UDP/IP-basierte Multicastpakete sendet, diversen Empfängerprogrammen, welche diese empfangen (vgl. Abbildung 5) sowie Auswertungen bzw. Statistiken über den Empfang erstellen. Dabei wurde die verfügbare Bandbreite vollständig ausgenutzt.

Abbildung 5: Das Analyseszenario. Zum Mithören der Paketrahmen in einem Netzwerksegment wurde ein Tap – ein aktiver bzw. passiver Splitter eines Ethernetbuskabels – und ein an diesem gekoppelter Netzwerk-Hardwareanalysator an den Positionen (A) oder (B) mit dem Netzwerk verbunden.

Auch bei den Testprogrammen traten Datenverluste, die bereits beim Monitor-Clients beobachtet wurden, auf. Quantitativ änderte sich die Verlustrate in Abhängigkeit von der

Größe der Datenpakete (vgl. Abbildung 6; *send throughput* und *receive throughput*). Hohe Verlustraten gingen mit einer hohen CPU-Last einher.

Abbildung 6: Paketverluste in Abhängigkeit von der Nutzlast. Geringe Nutzlast der Pakete erzeugt große Verlustraten, die durch Optimierungen beim Empfänger verringert werden können.

6.2 Untersuchung von Datenverlusten beim Sender

Zunächst wurde geklärt, ob der Sender **alle** Datenpakete in der richtigen Reihenfolge sendet. Dazu wurde der Datenverkehr direkt hinter der Netzwerkkarte des Senders über einen Tap repliziert und in einen Netzwerk-Hardwareanalysator eingespeist (vgl. Abbildung 5, Position (A)). Sämtlicher Datenverkehr des Multicast-Senders wurde sowohl in das LAN als auch in das Analyse-Werkzeug transferiert. Anschließend konnte verifiziert werden, dass keine Datenverluste beim Sender auftreten. Jedoch wird die Aktivierung der Flusskontrolle entsprechend IEEE 802.3x bei einem Multicast-Sender und -Empfänger empfohlen (vgl. Abschnitt 6.3).

6.3 Untersuchung von Datenverlusten auf dem Netzwerk

Um mögliche Datenverluste auf dem Netzwerk zu untersuchen, kam wiederum der Netzwerk-Hardwareanalysator zum Einsatz, dieses Mal direkt vor der Netzwerkkarte des Empfängers (vgl. Abbildung 5, Position (B)). Die Analyseszenarien ließen den Schluss zu, dass keine Datenverluste auf dem Netzwerk auftreten, sofern das lokale Netzwerk folgende Bedingungen erfüllt:

- Keine Vermischung von Netzwerkkomponenten verschiedener Bandbreiten (z.B. 10 Mbit/s gemischt mit 100 Mbit/s)
- Keine Verwendung von Netzwerkkomponenten mit mangelhafter oder fehlender 802.3x Unterstützung

Mit dem Standard IEEE 802.3x wurde ein Flow-Control-Verfahren auf Ebene 2 des OSI-Referenzmodells (engl. Open Systems Interconnection Reference Model) spezifiziert. Dieses dient u.a. zur Vermeidung von Pufferüberläufen auf aktiven Netzwerkkomponenten sowie dem daraus folgenden Verwerfen von Rahmen.

6.4 Untersuchung von Datenverlusten beim Empfänger

Als letzte Ursache für die beobachteten Verluste blieb der Empfänger. Für diese Analyse-szenarien wurde der Empfänger mit Paketen geflutet. Da sichergestellt war, dass die Daten ohne Verluste an der Netzwerkkarte des Empfängers ankamen, mussten sie *irgendwo* zwischen der Netzwerkkarte und der Anwendungs-Schicht verloren gehen. Zur Untersuchung der Vorgänge im Betriebssystem wurde das quelltextoffene Betriebssystem Linux verwendet. Abbildung 7 zeigt die grundsätzlichen Verarbeitungsstufen beim Empfang von Netzwerkpaketen eines Rechnersystems mit Linux als Betriebssystem.

Abbildung 7: Verarbeitungsstufen beim Empfang von Netzwerkpaketen unter Linux nach [6]

6.4.1 Empfangs-Verlustursache 1: Interrupts der Netzwerkkarte

Das Verhalten eines Empfängers ist abhängig von der Nutzlast bzw. Rahmengröße der empfangenen Multicastpakete (vgl. Abbildung 6). Durch den Empfang von vielen auf UDP basierten Paketen mit kleiner Nutzlast entsteht ein temporärer Engpass in der UDP-Kommunikation, aus dem Datenverluste resultieren. Bei einer zusätzlich künstlich geschaffenen CPU-Last erhöht sich der Datenverlust. Durch die Aufschlüsselung der Verteilung einer beanspruchten CPU (engl. Central Processing Unit) auf System-, Benutzer- und IRQ-Last (engl. Interrupt Request) sowie die Protokollierung erzeugter Interrupts wurde das massive Auslösen von Interrupts seitens der Netzwerkkarte infolge einer großen Anzahl eingehender Netzwerkpakete als Quelle der Verluste identifiziert. Die privilegierte Ausführung der Interrupt-Routine verbraucht dabei einen Großteil der System-Ressourcen, so dass die verbleibenden Ressourcen für die Auslieferung der Netzwerkpakete zur Anwendung nicht ausreichen. Die dadurch entstehenden Paketverluste sind signifikant.

Durch geeignete Treiberkonfigurationen konnte eine Drosselung der Interruptrate erreicht werden. Anstatt einer asynchronen Benachrichtigung über empfangene Netzwerkpakete instruiert der Treiber die Netzwerkkarte in regelmäßigen zeitlichen Abständen, einen Interrupt zu generieren. (Bei der Treibereinstellung des `e1000`-Moduls einer intel® Netzwerkkarte wurde mit einer Einstellung der ITR (engl. InterruptThrottleRate) auf 2.500 die besten Ergebnisse erzielt.) Damit konnte die CPU-Last drastisch reduziert werden. Dieses führte zu einer markanten Verringerung der Verluste (vgl. Abbildung 6 *Receive Throughput ITR optimized*).

Nun treten die Datenverluste nicht kontinuierlich auf, sondern impulsartig. Im Anschluss an eine Verlustphase steigt die Datenempfangsrate wieder sprunghaft an. Es gibt also eine weitere Ursache für Paketverluste bei einem verarbeitenden Mechanismus im Linux Netzwerk-Protokollstapel. Ein hardwarebedingter Paketverlust kann somit ausgeschlossen werden.

6.4.2 Empfangs-Verlustursache 2: Der Netzwerk-Stack und seine Puffer

Innerhalb einer Netzwerk-Anwendung im Userspace werden BSD-Sockets (vgl. Berkeley Software Distribution) als Schnittstelle zwischen dem Kernel- und dem Userspace verwendet [7]. Diese BSD-Sockets kommunizieren mit kernelinternen Sockets über die sog. Systemaufrufe.

Jeder BSD-Socket und somit auch jeder durch diesen assoziierte kernelinterne Socket besitzt seinen eigenen Sende- und Empfangspuffer. Die Größe der Sende- und Empfangspuffer fällt kernel- und distributionsspezifisch unterschiedlich aus und bezieht sich immer auf die Größe der Struktur, die zur »Aufnahme« bzw. zur Verarbeitung eines Netzwerkpaketes im Kernel dient. Beide Größen können mittels der Funktion `setsockopt()` konfiguriert und sollten für den vorliegenden Anwendungsfall maximiert werden. Die Standard- sowie die Maximalwerte der Puffergrößen eines kernelinternen Sockets sind beispielsweise über das `proc`-Dateisystem manipulierbar. (vgl. Dateieinträge im Verzeichnis `/proc/sys/net/core`)

Empfangspuffer werden sowohl von TCP als auch von UDP zur Aufnahme ankommender Netzwerkpakete verwendet. Durch die im Protokoll TCP spezifizierte Flusskontrolle kann der Empfangspuffer eines auf TCP basierten Sockets nicht zum Überlaufen gebracht werden, da es dem Kommunikationspartner nicht erlaubt ist, über das ihm bekannte Fenster hinaus weitere Netzwerkpakete zu senden. Ignoriert der Kommunikationspartner das ihm bekannt gegebene Fenster und überträgt mehr Daten bzw. Netzwerkpakete, werden die empfangenen Netzwerkpakete kontrolliert gelöscht.

Anders verhält es sich bei UDP, das keine Flusskontrolle besitzt. Die Größe der Empfangspuffer ist der Quelle unbekannt. Ein Datagramm, das nicht in den Socket-Empfangspuffer passt, wird gelöscht. Ein schneller Sender oder kurzzeitig auftretende Netzwerkpitzen können demzufolge einen langsameren Empfänger mit Netzwerkpaketen überfluten. Um dieses Verhalten nachzuweisen, wurde ein spezielles Kernelmodul zur Laufzeit in den Linux-Kernel geladen. Dieses protokolliert Angaben über die Füllzustände der Puffer und erlaubt den Abruf dieser Daten über das `proc`-Dateisystem. Der Vorgang des Verwerfens empfangener Multicastpakete konnte somit in der Transportschicht bei der UDP-Verarbeitung im SOCK RCV-Puffer (vgl. Abbildung 7) nachgewiesen werden.

Mit der Beseitigung dieser Engpässe bei den Empfänger-Rechnersystemen können die Datenverluste bei der Verwendung von UDP vollständig beseitigt werden. Testdurchläufe mit entsprechend optimierten Systemen ergeben UDP-Verlustraten, die unter normaler Last im lokalen Netzwerk deutlich unter einem Paket von zehntausend gesendeten Paketen liegen, was für das Monitoring-System als akzeptabel gilt.

7. Zusammenfassung

Die Anzahl an Echtzeit-Multicast-Anwendungen unter der Verwendung von IP-Multicast und UDP wächst ständig. Das vorgestellte Monitoring-System ist eine davon.

Obwohl UDP *unreliable* ist, werden zumindest in lokalen Netzen keine Datenverluste erwartet, solange der Datenverkehr nicht in die Größenordnung der Bandbreitenkapazität der beteiligten Netzwerkkomponenten kommt. Wie die vorliegende Untersuchung zeigt, gibt es jedoch Datenverluste bereits bei wesentlich kleineren Datenraten, die unter nicht-optimalen Bedingungen am Empfänger auftreten. Dort können zu viele Interrupts die CPU blockieren und im Protokollstack des Betriebssystems können Puffer überlaufen. Beides führt zum Verwerfen von Paketen.

Überschreitet die Datenrate einen kritischen Bereich, werden mitunter so viele Pakete verworfen, dass absolut weniger Daten ihr Ziel erreichen als bei geringeren Datenraten, bei denen es zu keinen Verlusten kommt.

Dieses ist auch eine Ursache des Fehlverhaltens Software-basierter Analyse-Werkzeuge für lokale Netzwerke.

Um den Nachteil der Unzuverlässigkeit von UDP auszugleichen, existieren diverse Netzwerk-Protokolle. Sie versuchen, die durch UDP auftretenden Verluste zu erkennen und ggf. durch Quittungen, Timeouts, Übertragungswiederholungen und die Verringerung der Sendebandbreite zu beseitigen. Dabei werden das Netzwerk sowie Sender und Empfänger zusätzlich belastet. Kritisch ist in diesem Zusammenhang, dass ggf. ein bereits im Empfänger befindliches Netzwerkpaket erneut vom Kommunikationspartner angefordert wird, weil es verworfen wurde. Mit solchen Protokollen wird das Symptom »Datenverlust« behandelt, nicht jedoch dessen Ursache beseitigt.

8. Danksagung

Herzlicher Dank gilt der XDV- und der Steuerungsgruppe am IPP sowie den Betreuern der Masterarbeit Herrn Prof. Stütz und Frau Prof. Falkner von der FH Stralsund.

9. Literatur

- [1] Tambach, Steffen: Analyse eines verbindungslosen Netzwerkprotokolls der Gruppenkommunikation in Ethernet-basierten Netzwerken, Masterarbeit M.Sc., Fachhochschule Stralsund, 2006
- [2] Hohpe, G.: Programmieren ohne Stack: Ereignis-getriebene Architekturen; in ObjektSpektrum 2/2006
- [3] IHK Abschlussprüfung Sommer 2006 Fachinformatiker/Fachinformatikerin Systemintegration 1197 1 Ganzheitliche Aufgabe I Fachqualifikationen 2. Handlungsschritt Frage b (ZPA FI Ganz I Sys 5)
- [4] <http://www.ipp.mpg.de/ippcms/de/pr/exptypen/stellarator/plasmadiagnostik/index.html>
- [5] Hennig, Christine: A Concept of Online Monitoring for the Wendelstein 7-X Experiment, Fusion Engineering and Design, Volume 71, Issues 1-4, Pages 107-110
- [6] Wu, W., Crawford, M.: The performance analysis of Linux networking – packet receiving. http://cd-docdb.fnal.gov/0013/001343/001/CHEP06_Paper_132_Wu-Crawford_LinuxNetworking.pdf, [2006-04-12].
- [7] Stevens, W. Richard: UNIX Network Programming. Prentice Hall Software Series, Prentice-Hall, Inc. A Simon & Schuster Company Englewood Cliffs, New Jersey 07632, 1990.