Integratives Maßschneidern einer Kommunikationsmiddleware für eine Luftschiffapplikation

Daniel Schneider, Alexander Geraldy

{geraldy, d_schnei}@informatik.uni-kl.de
AG Vernetzte Systeme, TU Kaiserslautern
67663 Kaiserslautern, Deutschland

Abstract: Dieser Beitrag beschreibt das Maßschneidern einer Kommunikationsmiddleware für eine Luftschiffanwendung. Die Middleware unterstützt die selbstorganisierende und mobile Kommunikation innerhalb eines Ad-Hoc-Netzwerks und stellt als Hauptfunktionalitäten ein reaktives Unicast- und Multicast-Routingverfahren sowie Mittel zur Gewährleistung der zuverlässigen Steuerung des Luftschiffs zu Verfügung. Der Prozess des Maßschneiderns gliedert sich in Auswahl des Routingverfahrens und systematischen Entwurf und wird besonders hinsichtlich dessen integrativer Aspekte beleuchtet. Ziel der Integration ist die bestmögliche Anpassung an die Anforderungen der Anwendung unter effizienter Nutzung des Mediums.

1 Einleitung

Mobile Adhoc-Funknetze bieten eine Fülle an Optionen für zukünftige Anwendungen – wie z.B. im Bereich *Ambient Intelligence* oder im digitalen BOS[1]-Funksystem. Bei der Digitalisierung des BOS-Funks existieren harte Anforderungen an die Funktionalitäten, die Systemqualität und ausgewählte Dienstgüteaspekte, um beispielsweise auch Katastrophenlagen bewältigen zu können. In diesen Situationen spielt die Selbstorganisation und die Skalierbarkeit des Netzes, sowie die Einhaltung der Dienstgüte eine übergeordnete Rolle. Die Dienstgüte kann hierbei i. d. R. nicht durch generische Lösungen und ein Überangebot an Ressourcen bewältigt werden, sondern nur durch Protokolle, welche auf die Anforderungen zugeschnitten sind. Hierbei sind im Idealfall sowohl der vollständige Entwurfsprozess als auch alle Schichten des Kommunikationssystems maßgeschneidert; so können die Protokolle von einer horizontalen und vertikalen Integration profitieren. Ein weiterer wichtiger Aspekt ist der korrekte Entwurf des Systems, da die Systemqualität unabdingbar ist. Eine formale Spezifikation des Systems sowie der Einsatz von Wiederverwendungsansätzen kann die Systemqualität elementar verbessern.
Wir gehen im Folgenden auf die Spezifikation einer Middleware mit Routingfunktionalität für den Luftschiffdemonstrator ein. Besonderes Augenmerk ist hierbei auf das Maßschneidern und die Integration gerichtet. Aus dem komplexen Bereich des Rettungswesens be-

[1] BOS: Behörden und Organisationen mit Sicherheitsaufgaben

trachten wir eine spezielle Applikation: Den Einsatz einer ferngesteuerten Drohne (hier ein Luftschiff; s. Abb. 1) zur Erkundung und Überwachung eines Schadensgebietes mit Hilfe eines Videosystems, sowie die Verwendung der Drohne als mobiler Router.

Abbildung 1: Luftschiff-Demonstration

2 Anforderungen der Anwendung

Bei der Entwicklung des Luftschiffdemonstrators rücken zwei Anwendungen in den Vordergrund: Zum einen die Steuerung des Luftschiffs (PILOTSERVER und PILOTCLIENT), zum anderen die Aufzeichnung, Übertragung und Anzeige von Videodaten einer in die Gondel integrierten Webcam (VIDEOSERVER und VIDEOCLIENT). Die beiden Serverapplikationen sind auf dem Luftschiff angesiedelt und haben die Ausführung der Steuerkommandos bzw. das Senden von Videodaten zur Aufgabe. Ferner soll der PILOTSERVER allen beteiligten Steuergeräten regelmäßig Feedback mit den aktuellen Steuerparametern zukommen lassen. Die Client-Applikationen befinden sich auf den mobilen Geräten und haben das Steuern des Luftschiffs und die Darstellung der empfangenen Videodaten zur Aufgabe. So ergeben sich folgende grundlegende Kommunikationstypen und Anforderungen:

- Die **Übertragung der Steuerkommandos** (z.B. Drehzahl des Heckrotors und der Auftriebsrotoren, Servostellung) erfolgt von den einzelnen mobilen Steuergeräten zum Luftschiff und lässt sich mit Unicastverbindungen realisieren. Die Feststellung eines Verbindungsabbruchs ist elementar, um im Notfall das Steuerrecht neu zu vergeben oder den Zeppelin in einen Fail-Safe-Zustand zu überführen.
- Die **Übertragung des Steuerfeedbacks** (z.B. aktuelle Motordrehzahlen, Servostellung, Ladezustand des Akkus) erfolgt vom Luftschiff zu allen Steuergeräten. Diese Multicast-Verbindungsstruktur lässt sich entweder durch ein Multicastverfahren realisieren oder kann durch mehrere Unicastverbindungen nachgebildet werden kann.

- **Übertragung der Videodaten**
- **Allgemeine Anforderungen** an das Routingverfahren sind ein möglichst geringes Gesamtvolumen des Kontrollverkehrs, eine gute Skalierbarkeit (vor allem bezüglich der Anzahl der Empfänger) und eine gute Erweiterbarkeit.

3 Maßschneidern der Kommunikationsmiddleware

3.1 Das Routingverfahren

Zur Erfüllung der Anforderungen bieten sich sowohl Unicast- als auch Multicastverfahren an. Wir gehen nun kurz auf Vertreter dieser Typen ein und wählen schließlich ein geeignetes Verfahren aus.

3.1.1 Unicast- und Multicast-Routing

Unicastverfahren lassen sich anhand ihrer Ermittlung des Netzzustandes, ihrer Sicht auf das Netz und der Art ihrer Leitwegermittlung unterscheiden. Die Ermittlung des Netzzustandes kann *proaktiv* und *reaktiv* erfolgen. Typische Sichten sind *global-aggregiert*, *partiell-aggregiert*, *partiell* und *lokal*. Leitwege können *quellbasiert*, *verteilt* oder *hierarchisch* ermittelt werden. Beispiele für Unicastverfahren sind DSDV [1] (proaktiv, global-aggregiert, verteilt), CGSR [2] (proaktiv, partiell-aggregiert, hierarchisch), LANMAR [3] (proaktiv, partiell, verteilt), AODV [4] (reaktiv, partiell aggregiert, verteilt), DSR [5] (reaktiv, partiell, quellbasiert) und ZRP [6] (proaktiv/reaktiv, partiell, hierarchisch).
Im Bereich von Multicast-Routing existieren baumbasierte und meshbasierte Verfahren. Bei den baumbasierten Verfahren kann man zwischen quellbasierten und kernbasierten Varianten unterscheiden. Bei den meshbasierten Verfahren existiert der Sonderfall der gruppenbasierten Verfahren, die nicht Verbindungen verwalten, sondern sogenannte Weiterleitungsknoten. Beispiele für Multicastverfahren sind DVMRP [7] (quellbasierte Bäume, proaktiv), ABAM [8] (quellbasierte Bäume, reaktiv), AODV-Multicast [9] (kernbasierte Bäume, reaktiv), CAMP [10] (meshbasiert, proaktiv) und ODMRP [11] (gruppenbasierte Weiterleitung, reaktiv).

3.1.2 Auswahl des Routingverfahrens

Im Kontext des vorliegenden Szenarios entstünde durch die Nachbildung des benötigten Multicast durch Unicast ein erheblicher Overhead. Dies schränkt die potentiell geeigneten Verfahren auf die Klasse der Multicastverfahren ein. Grundsätzlich erscheint zur Verwendung in mobilen Szenarien ein meshbasierter Ansatz von Vorteil. Hier stehen beim Ausfall eines Links im günstigen Fall alternative Leitwege zur Verfügung, wodurch aufwändige Rekonfigurationen der Multicaststruktur entfallen ([12, 13]).
Hinsichtlich der Anforderungen des Demonstrators erscheint ODMRP (On Demand Mul-

ticast Routing Protocol) besonders geeignet, es basiert auf der Struktur der Weiterleitungsgruppe. Seine Stärken liegen im vergleichsweise geringen Kontroll- und Speicheroverhead, der Robustheit bezüglich Mobilität durch die implizite Nutzung alternativer Pfade, die Ausnutzung der Broadcasteigenschaft drahtloser Netze und die gute Skalierbarkeit bezüglich der Anzahl der Multicastempfänger. Zusätzlich werden wir aufzeigen, dass ODMRP im vorliegenden Szenario mittels einer Erweiterungen (die Verwendung der Rückwärtspfade) eine zusätzliche Optimierung ermöglicht (siehe Kapitel 4.2).

3.1.3 Funktionsweise von ODMRP [11]

Ein Sender initiiert eine Multicastsession, indem er ein sogenanntes JOINREQUEST als Broadcast-Nachricht an seine Nachbarknoten sendet. Jeder Knoten aktualisiert beim ersten Erhalt dieses Pakets seine Routingtabelle, trägt den Vorgängerknoten ein und sendet dann das Paket weiter (→Fluten). Möchte ein Knoten Mitglied der Multicastgruppe werden, sendet er ein sogenanntes JOINCONFIRM aus. Dieses enthält die Adresse des Multicastsenders und des Vorgängerknotens, bekannt durch den empfangenen JOINREQUEST. Da ein Empfängerknoten gleichzeitig Mitglied in verschiedenen Multicastgruppen sein kann, werden die Informationen über Multicastsender und Vorgängerknoten als Tabelle (*JoinTable*) verwaltet. Bei Erhalt eines JOINCONFIRM überprüft der betreffende Knoten, ob seine Adresse als nächster Hop zu einem (oder mehreren) der Multicastsender in der JoinTable enthalten ist. Ist dies der Fall, wird der nach lokalen Routinginformationen nächste Hop in Richtung des betreffenden Multicastsenders eingetragen und die Nachricht weitergeleitet. So gelangt das JOINCONFIRM auf dem nach Best-Effort-Gesichtspunkten kürzesten Weg vom Empfänger zu allen zugehörigen Multicastsendern. Alle Knoten, welche das JOINCONFIRM weitergeleitet haben, werden Weiterleitungsknoten für die betreffende(n) Gruppe(n). Dies bedeutet, dass sie von nun an alle eingehenden Nachrichten der zugehörigen Sender weiterleiten. Um die Aktualität der Weiterleitungsgruppe zu gewährleisten, werden JOINREQUESTS und JOINCONFIRMS periodisch gesendet. Das Verlassen einer Multicastgruppe erfolgt durch das Unterlassen der JOINCONFIRMS, das Ende einer Multicastsession erfolgt entsprechend durch das Unterlassen der JOINREQUESTS durch den Sender, die Multicastverbindungen sind also nach dem *Soft-State*-Prinzip realisiert.

3.2 Entwicklungsprozess

Wir beschreiben im folgenden den iterativen Entwicklungsprozess, welcher ausgehend von den Applikationen über einzelne Verfeinerungsschritte zum endgültigen System hinführt; durch die Aufteilung der Funktionalitäten auf die verschiedenen Iterationen ist eine bessere Beherrschbarkeit der Komplexität gegeben. Innerhalb einer Iteration werden alle zentralen Funktionalitäten aus den Anforderungen zunächst durch MSCs (Message Sequence Charts) beschrieben, um so die Abbildung auf SDL (Specification and Description Language, [14]) vorzubereiten. Durch dieses Vorgehen wird eine gute Verfolgbarkeit im Entwicklungsprozess, zum einen zwischen Anforderungen, MSCs und SDL und zum anderen zwischen den Produkten der Verfeinerungsschritte, erreicht. Zur Erstellung von

MSCs, SDL-Spezifikationen, und Code, sowie zur Simulation wird *Telelogic Tau 4.5* eingesetzt, ein kommerzielles Werkzeug zur Entwicklung verteilter Systeme und Kommunikationsprotokolle.
Der konkrete Entwicklungsprozess gliedert sich in drei Schritte, die sich durch eine zunehmende Verfeinerung des Systems bei gleichzeitiger Abnahme der Abstraktion auszeichnen. Der erste Schritt umfasst die *Spezifikation der Applikation*. Ihre Aufgabe umfasst zusätzlich zur eigentlichen Steuerung des Luftschiffs die Initiierung und Beendung der Multicast-Sitzung sowie die Kontrollvergabe innerhalb der Gruppe der möglichen Steuergeräte. Diese Kontrollvergabe ist ein Beispiel für die Integration von Uni- und Multicast, auf die in Kapitel 4 eingegangen wird. Das Routing ist aufgrund der hohen Abstraktion kein Gegenstand dieses Entwicklungsschrittes.
Die *Entwicklung der Middleware* beinhaltet zum einen die Integration des Routingverfahrens (nicht applikationsspezifisch) und zum anderen die Umsetzung der applikationsspezifischen Funktionalitäten (zum Beispiel die Verwaltung einer Multicastsession und die Umsetzung des Watchdog-Heartbeat-Mechanismus, siehe Kapitel 4.2). Eine entsprechende Unterteilung der Middleware in applikationsspezifische und nicht-applikationsspezifische Teile bietet sich folglich an. Nach der Durchführung dieser Iteration ist die Simulation des Routingverfahrens kombiniert mit den Funktionalitäten der Applikationen möglich.
Zunächst wird noch davon ausgegangen, dass genau eine Applikation auf jedem physikalischen Knoten existiert. Diese Einschränkung wird in der *Partitionierung des Systems* aufgehoben, in der durch geeignete Erweiterung der Adressierung und Paketzustellung mehrere Applikationen auf einem Host ermöglicht werden.

4 Integrative Aspekte des Maßschneiderns

4.1 Integration von Unicastrouting

Entsprechend der Anforderungen der Applikationen wird sowohl Unicast- als auch Multicastkommunikation benötigt. Wie bereits beschrieben handelt es sich bei ODMRP um ein Multicast-Routingverfahren. Grundsätzlich wäre es aber auch möglich, ODMRP als Unicastverfahren einzusetzen. Alle Multicastgruppen hätten in diesem Falle lediglich ein einziges Mitglied. Hier ergibt sich allerdings das Problem eines hohen Kontrolloverheads, da jeder Unicastsender das Netz mit seinen periodischen JOINREQUESTS flutet.
Im Zuge des Maßschneiderns ist eine bessere Lösung zu finden: Unicastkommunikation findet im Kontext des Luftschiffdemonstrators nur zwischen den PILOTCLIENTS und dem PILOTSERVER statt. Alle PILOTCLIENTS sind aber bereits Mitglied der Multicastgruppe des Pilotservers. Somit sind die zum Unicast benötigten Routinginformationen durch eine gezielte Ausnutzung des Multicast-Kontrollverkehrs ohne zusätzliche Kontrollnachrichten zu erlangen. Durch den Austausch der JOINREQUESTS und JOINCONFIRMS zwischen dem Multicastsender und den Multicastempfängern werden also gleichzeitig die *bidirektionalen Leitwege für Unicast* erstellt und aktualisiert.

4.2 Steuerfeedback und Kontrollvergabe

Als besonderer Aspekt des Maßschneiderns ist die Integration von Unicast und Multicast im Kontext des Steuerfeedbacks und der Kontrollvergabe hervorzuheben. Das Senden der Steuerkommandos vom kontrollierenden PILOTCLIENT zum PILOTSERVER erfolgt per Unicast, während das Senden des Steuerfeedbacks (nach jedem Steuerkommando) vom PILOTSERVER an alle teilnehmenden PILOTCLIENTS per Multicast erfolgt. Das Steuerfeedback erfüllt nun zwei integrierte Aufgaben: Zum einen soll es stets alle teilnehmenden PILOTCLIENTS über die aktuellen Steuerparameter des PILOTSERVERS informieren. Zum anderen erhält der aktive PILOTCLIENT eine wichtige Bestätigung seiner Steuerinformation, ohne dass eine weitere Unicast-Verbindung vom PILOTSERVER zum aktiven PILOTCLIENT aufgebaut wird. Wie bereits erläutert werden für den Unicast der Steuerkommandos die bereits durch den Multicast des Feedbacks etablierten bidirektionalen Pfade verwendet. Somit ist der Aufbau von Leitwegen über die Multicastsession des Feedback hinaus nicht notwendig.
Der Kontrollvergabemechanismus besitzt die Struktur eines Three-Way-Handshakes. Zunächst signalisiert der PILOTSERVER die Übergabebereitschaft per Multicast an alle PILOTCLIENTS. Nun steht es den PILOTCLIENTS frei, sich per Unicast beim PILOTSERVER um die Kontrolle zu bewerben. Der PILOTSERVER teilt daraufhin – wiederum per Multicast – mit, welcher PILOTCLIENT von nun an die Kontrolle besitzt. Auch dieser Handshake setzt also auf der Integration von Unicast- und Multicast-Verbindungen.

4.3 Integration einer Watchdog-Funktionalität

Für die Sicherheit des Luftschiffes ist es unerlässlich, einen Verbindungsverlust zum PILOTCLIENT möglichst bald zu bemerken und gegebenenfalls den Kontrollvergabemechanismus zu initiieren. Ferner könnten in einem solchen Fall weitere Maßnahmen, wie der Wechsel in einen fail-safe oder fail-operational Zustand, ergriffen werden. Um einen Verbindungsausfall erkennbar zu machen, sendet der PILOTCLIENT die aktuellen Steuerkommandos (Unicast, unreliableSend(com1), siehe Abb. 2) periodisch bis zu deren Bestätigung (Multicast, unreliableSend(fdb1 ...)) an den PILOTSERVER. Anschließend wird mit dem ebenfalls periodischen Senden von Heartbeat-Signalen (unreliableSend(heartbeat)) begonnen. Kommandos und Heartbeats führen im PILOTSERVER zu einem Rücksetzen eines Watchdog-Timers. Erst bei Verlust einer festgelegten Anzahl von Heartbeats läuft der Watchdog-Timer ab (in Abb.2 nicht dargestellt) und initiiert eine neue Kontrollvergabe. Durch diese Zweistufigkeit kann erreicht werden, dass verlorene Steuerkommandos zügig wiederholt werden, während nach dem Erhalt des zugehörigen Feedbacks die Datenrate drastisch reduziert werden kann. Die Ergänzung dieser Funktionalität erfolgt durch die Anwendung der Heartbeat- und Watchdog-Patterns[16], die zum einen eine vorgefertigte Lösung für dieses Problem beschreiben, und zum anderen durch strukturierte Anleitungen und Checklisten die Qualität des Entwurfs sicherstellen.

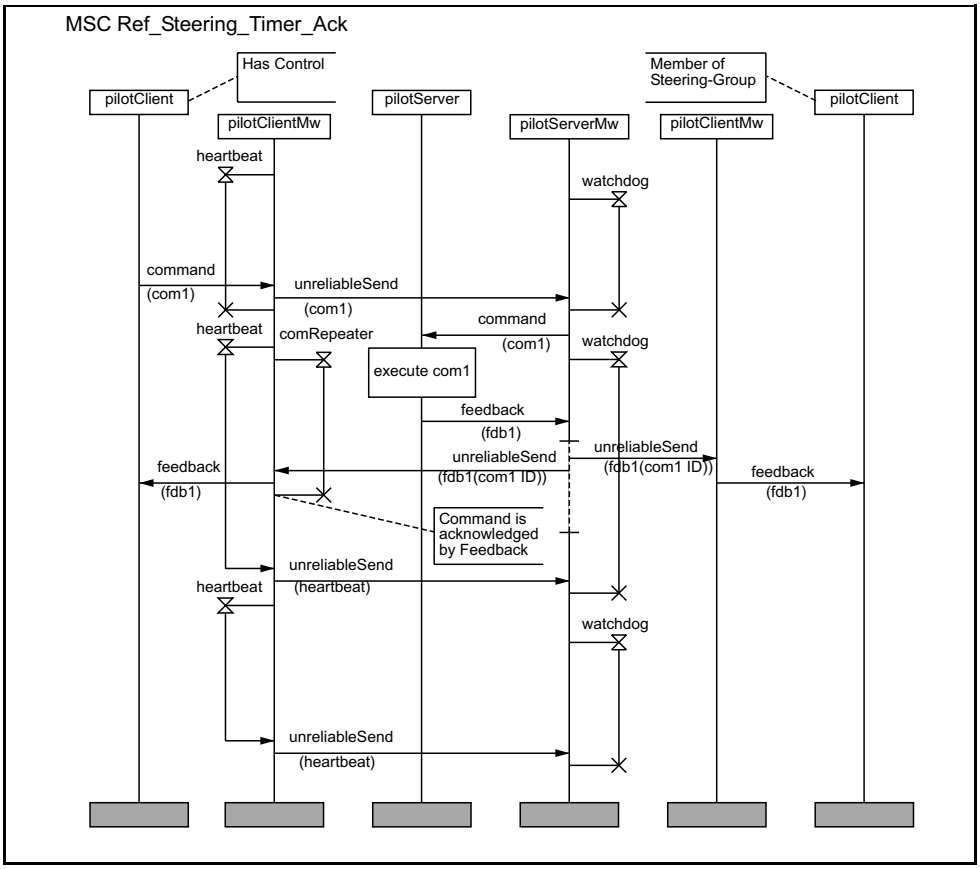

Abbildung 2: Heartbeat und Watchdog, implizite Bestätigung durch Feedback

5 Ausblick

Die dargestellte Middleware bietet eine Vielzahl an Erweiterungsmöglichkeiten. Hierunter spielt die Integration von QoS-Funktionalitäten – insbesondere bzgl. des Routings und des Datentransports – eine bedeutende Rolle. Arbeiten zum Maßschneidern einer geeigneten Architektur sind zur Zeit im Gange. Ergebnisse sind hier vor allem bei der Untersuchung der horizontalen und vertikalen Integrationsmöglichkeiten zur Basistechnologie denkbar. Die weiteren Arbeiten haben den Aufbau bzw. die Erweiterung eines Pattern-Pools und einer Mikroprotokollbibliothek [17] zum Ziel. Beide bieten das Potenzial, zukünftiges Maßschneidern zu beschleunigen und qualitativ anzuheben.

Literatur

[1] C. E. Perkins and P. Bhagwat: *Highly Dynamic Destination Sequenced Distance Vector Routing (DSDV) for Mobile Computers;* ACM SIGCOMM, pp. 234–244, October 1994.

[2] Ching-Chuan Chiang, Hsiao-Kuang Wu, Winston Liu, and Mario Gerla: *Routing in clustered multihop, mobile wireless networks with fading channel;* Proceedings of IEEE Singapore International Conference on Networks (SICON '97), pp. 197–211, April 1997.

[3] G. Pei, M. Gerla and X. Hong: *LANMAR: Landmark Routing for Large Scale Wireless Ad Hoc Networks with Group Mobility;* MobiHOC'00, 2000.

[4] C.E. Perkins and E.M.Royer: *Ad-hoc on demand distance vector routing;* in Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, February 1999, pp. 90–100.

[5] David B. Johnson and David A. Maltz: *Dynamic source routing in ad hoc wireless networks;* In Mobile Computing; Kluwer Academic Publishers, 1996.

[6] Z. Haas: *A new routing protocol for the reconfigurable wireless networks;* IEEE Conf. on Universal Personal Comm., pp. 562–566, '97.

[7] Waitzman, Partridge, Deering: *Distance Vector Multicast Routing Protocol;* Stanford University, RFC 1075, November 1988.

[8] C.-K. Toh, G. Guichal, S. Bunchua: On-demand associativity-based multicast routing for ad hoc mobile networks (ABAM), Vehicular Technology Conference, 2000. IEEE VTS Fall VTC 2000. 52nd, Volume: 3, 2000, Page(s): 987–993 vol.3

[9] E. M. Royer and C. E. Perkins: *Multicast Ad hoc On-Demand Distance Vector (MAODV) Routing;* In Proc. of the Second IEEE Workshop on Mo-bile Computing Systems and Applications, 1999.

[10] J.J. Garcia-Luna-Aceves and E.L. Madruga: *The Core-Assisted Mesh Protocol;* in IEEE Journal on Selected Areas in Communications, August 1999.

[11] S.-J. Lee, M. Gerla, and C.-C. Chiang: *Ondemand multicast routing protocol;* In Proceedings of IEEE Wireless Communications and Networking Conference (WCNC'99), September 1999.

[12] S.-J. Lee, W. Su, J. Hsu, M. Gerla and R. Bagrodia: *A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols;* Proceedings of IEEE INFOCOM 2000.

[13] S.-J. Lee, W. Su, and M. Gerla: *On-Demand Multicast Routing Protocol in Multihop Wireless Mobile Networks;* ACM/Baltzer Mobile Networks and Applications, special issue on Multipoint Communication in Wireless Mobile Networks, 2000.

[14] *ITU Recommendation Z.100 (11/99): Specification and Description Language (SDL);* International Telecommunication Union (ITU), Geneva, 2000

[15] D. Schneider: *Auswahl, Adaption und Entwurf eines Routingverfahrens für Ad-Hoc-Netzwerke;* Diplomarbeit, Technische Universität Kaiserslautern, Fachbereich Informatik, 2004

[16] C. Webel: *Entwicklung und Integration von QoS-Mikroprotokollen zur Steuerung eines Fluggerätes über WLAN;* Diplomarbeit, Technische Universität Kaiserslautern, Fachbereich Informatik, 2004

[17] R. Gotzhein, F. Khendek, P. Schaible: *Micro Protocol Design: The SNMP Case Study;* Proc. of 3rd SAM (SDL And MSC) Workshop, 2002