Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Einführung

Der Einsatz von mobilen Endsystemen nimmt heute stark zu und wird zunehmend in bestehenden Datennetzen, die ihrerseits auch zunehmend integriert und vernetzt werden, eingebracht. Die physische Architektur von Mobilsystemen, diese umfassen zellulare WANs, drahtlose LANs, Satellitensysteme, Pagingnetze sowie auch Festnetze, ist im Gegensatz zu Festnetzen sehr dynamisch. Die Endgeräte sind durch geringe Speicher-, CPU- und Energiekapazitäten charakterisiert, und dazu kommt eine stark variable Bandbreite. Daraus entstehen neue Managementfunktionen:

1. Location Management,
2. Power Management,
3. und Bandbreiten Management,

neben den Standard-Funktionen: Konfigurationsmanagement, Fehlermanagement, Sicherheitsmanagement, Leistungsmanagement und Abrechnungsmanagement, die in bestehenden Managementsystemen wie in [HA93] beschrieben ist, integriert werden sollen.
Eine der wichtigsten Anwendungen von Mobilsystemen ist Messaging (Electronic Mail, Voice-Messaging, Paging). Ein großer Teil der Privat- und der Geschäftskommunikation läuft heute über Email, Paging und sonstige Messagingsysteme. Ein Mobilbenutzer soll heute in der Lage sein, einerseits Nachrichten unabhängig von Zeit und Aufenthaltsort unter Berücksichtigung der Zugriffsrechten und Hardwaregrenzen empfangen und verschicken zu können, anderseits zusammengesetzte Messagingsysteme (z.B. Voice und Email) ohne großen Aufwand oder detaillierte fachliche Kenntnisse leicht und transparent zu nutzen.

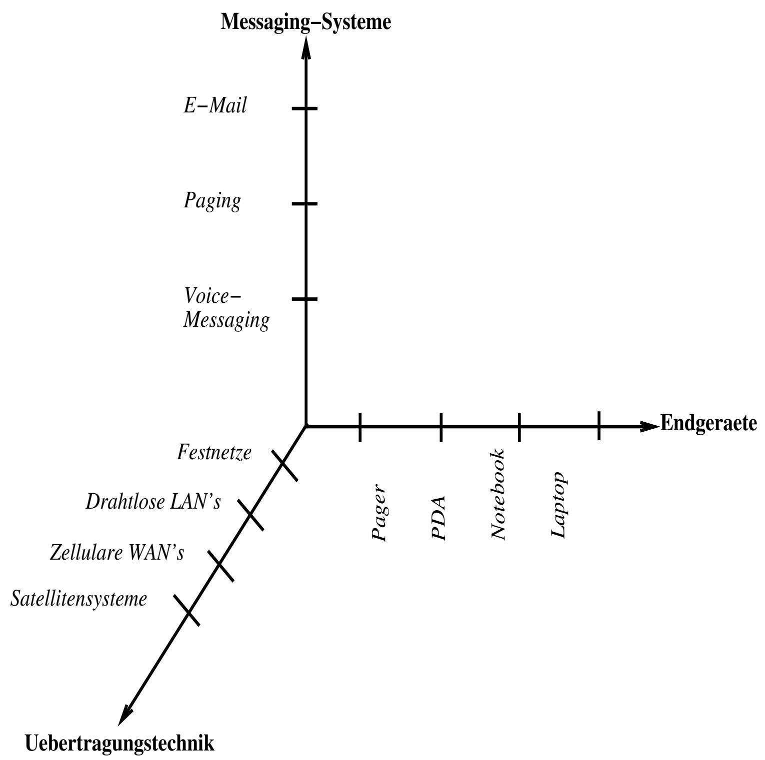


Abbildung 1.1: Mobilsysteme

Netzmanager haben es heute nicht nur mit einem täglichen Nachrichtenvolumen von mehreren Gigabytes zu tun, sondern auch mit komplizierten Managementaufgaben bedingt durch die Einbringung von Mobilität, die viele neue Aspekte wie die Notwendigkeit des dynamischen Routings mit sich bringt, und die Heterogenität der Datennetze.
Die bestehenden Messagingprotokolle und deren Architekturen werden ständig weiterentwickelt, um wenn auch nur teilweise die Anforderungen der neuen Übertragungstechniken und Netzarchitekturen zu erfüllen. Es besteht daher der Bedarf nach einer umfassenden Untersuchung der Messagingprotokolle im Hinblick auf die Mobilität der Endgeräten und die Heterogenität der Netze, sowie Endgeräte. Die Festlegung einer eventuell geeigneten Messagingarchitektur und die Definition der Funktionalität ihrer Komponenten, sowie deren Kommunikationsbeziehungen ist notwendig, um eine Management-Lösung zu untersuchen, zu erweitern oder neu zu definieren.

1.2 Motivation

Messagingsysteme werden in einem breiten Umfeld eingesetzt. Im folgenden werden einige praktische Szenarien zur deren Nutzung angegeben, die in der Praxis zu treffen sind. Es wird die Mobilität der Endgeräte, die Heterogenität der Datennetze aufgezeigt. Die Möglichkeit, verschiedene Messagingsysteme zu integrieren, wird auch berücksichtigt.

1. *Heterogenität: Projekt-Team*
   Das erste Beispiel soll eine Softwareentwicklungsfirma sein. Das Entwicklungsteam dieser Firma ist im Firmengebäude ansässig und hat ein LAN-(TCP/IP) basiertes Email-System, laufend auf UNIX-Rechnern. Die Vertriebsmitarbeiter sind mit Laptops ausgerüstet, auf denen Windows-Email-Klienten installiert sind. Für Kunden ist eine Email-Hot-Line eingerichtet. Die unterschiedlichen Emailsysteme und Plattformen sollen möglichst einwandfrei funktionieren und miteinander harmonieren. Ohne ein Managementsystem ist so eine heterogene Umgebung schwer oder gar nicht verwaltbar.

2. *Konfiguration: Reisen*

   (a) Auf dem Weg zum Flughafen, egal ob privat oder geschäftlich, ist es meistens sinnvoll, Informationen über Parkplätze im Flughafen, Fluginformation (z.B.: Verspätungen) oder Reisewetter zu wissen, um gegebenenfalls andere Maßnahmen zu treffen. Solche Informationen sind z.B. mit einem Pager zu bekommen.

   (b) Ein reisender Mitarbeiter will Zugang zu seiner Mail haben. Dafür hat er einen Rechner mit Internet-Zugang im Büro, wo seine Mail ankommt, und ein Notebook mit PCMCIA-Modem, das er immer bei sich hat. Er beauftragt einen Provider, der ein Wirelessnetz betreibt, das alle Gebiete, wo er sich befinden kann, abdeckt, ihm seine Mails abzuspeichern und gegebenenfalls zuzustellen (zum Notebook).
   Wenn die Mail bei ihm ankommt, wird sie durch das Internet zur Privatfirma weitergeleitet (*Forward*), die Mails werden dann abgespeichert und nur auf sein Wunsch zugestellt.
   Der Mitarbeiter will nur die Mails von seinem Chef und seiner Frau lesen, er beauftragt dann die Firma in periodischen Zeitintervallen, die Nachrichtenköpfe zuzuschicken und wegen hohen Gebühren erst in der gebührenniedrige Zeitperiode die ausgewählten Mails zuzuschicken. Er selber will auch Nachrichten nur in dieser Periode verschicken (ausgenommen, die zu seinem Chef), aber er schreibt sie, wann er Zeit hat.

3. *Sicherheit*
   Messagingsysteme haben hohe Sicherheitsanforderungen, im folgenden werden einige Beispiele angegeben:

   (a) Systembetreuer wollen vertrauliche Benutzerdaten austauschen.

   (b) Firmenpläne und -Geheimnisse diskutieren und austauschen.

(c) Kundendaten, die unter dem Datenschutz-Gesetz fallen, transportieren.

4. *Mobilität: Außendienst, Service*
   Ein Mitarbeiter eines Lieferdienstes soll immer zu den Kunden fahren, er braucht auf dem Weg Informationen über Staus, Wetterlage, und günstige Verkehrsverbindungen. Radioinformationen reichen ihm nicht aus, weil sie in großen Zeitabständen, zu spät oder ungünstig gesendet werden, deshalb nutzt er einfach einen Pagingdienst (z.B.*ERMES, Scall, TelMe*). Um mit der Firma in Kontakt zu bleiben hat er ein Bordcomputer, womit er seine Momentane Position meldet und wodurch er Anweisungen von seinem Chef bekommt (z.B. mit Email, es gibt aber auch andere Verfahren (GPS,...)).

5. *Kombination verschiedener Messagingsysteme:*
   Mitarbeiter einer Firma, die mit Kollegen oder Kunden kommunizieren, haben nicht oft die Möglichkeit eine Echtzeitkommunikation durchzuführen, gerade wenn sie z.B. in einer Besprechung sind.
   Ein Mitarbeiter sollte aber flexibel und gut informiert sein. Es ist auch vom großen Vorteil, das er fast immer erreichbar ist. Ein Notebook mitzuschleppen um Mails abzuarbeiten ist in vielen Fällen unangenehm oder unrentabel z.B wenn kein Netzzugang möglich ist (fest oder mobil) oder wenn er sich in einem fremden Netz befindet und keine Zugriffsrechte hat. Öfter bietet sich das Telephon, das so gut wie überall vorhanden ist. Eine Alternative ist dann die Nachrichten mit dem Telephon anzuhören oder zu verschicken, was viele Vorteile mit sich bringt wie:

   (a) Entlastung des lokalen Netzes.

   (b) Der einfache Zugang zu einem Telephonnetz.

   Anhören von Nachrichten am Telephon stellt aber viele Anforderungen an das Messagingsystem, da es z.B keine Möglichkeit gibt die Nachrichtenköpfe anzusehen oder die Nachrichten wahlweise anzuhören. Es ist daher notwendig das Anhören von Nachrichten auf irgendeiner Art zu steuern.
   Der Mitarbeiter will z.B nur wichtige Nachrichten hören oder nur Eilnachrichten hören. Er kann auch wegen der mangelnde Sicherheit der Telephonnetze ein anders Medium auswählen, um vertrauliche Nachrichten zu lesen oder hören.
   Im folgenden ist eine Skizze zur Veranschaulichung dieses Szenarios an einem konkreten Produkt *Serenade* der amerikanichen Firma *Octel* angegeben.

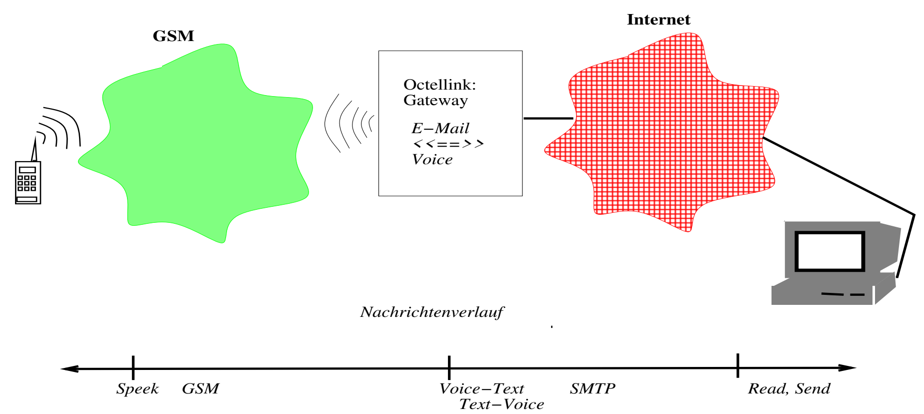


Abbildung 1.2: Email und Voice-Mail

6. *Nachrichtenvolumen*
   Die Menge an Information wie z.B in WWW-Seiten oder im Internet ist gestiegen. Informationsaustausch ist selbstverständlich geworden. Allerdings sollen große Datenmengen ausgetauscht werden. Benutzer besonders im universitären Bereich schicken ihre Nachrichten mit eingebetteten Links auf Dokumenten, die dann auf anderen Wege geholt werden.

7. *Sharing*
   Ein Team, das an einem Projekt arbeitet, teilt sich eine Mailbox um Informationen auszutauschen, auch wenn sie sich örtlich bewegen und weit voneinander getrennt sind.

8. *Konferenzen und Meetingsitzungen*
   Die Teilnehmer einer Konferenz, die Messaginganwendungen benutzen wollen, sollen entweder ein Zugang zum lokalen Netz haben unter Berücksichtigung ihrer Rechte zur Nutzung von lokalen Messaginganwendungen, oder über Funknetze einen Zugang zum Heimatnetz beschaffen (z.B *GSM*[1]). Die ausgetauschte Nachrichten sind meistens kurz und müssen schnell und möglichst fehlerfrei ankommen. Es können bei Konferenzen und Meetingsitzungen verschiedene Probleme identifiziert werden:

   (a) Funkverbindungen haben eine große Fehlerrate (verursacht meistens durch Funkschatten)

[1] Global Systems for Mobile communication

(b) Email ist zu zuverlässig. Nachrichten können verspätet ankommen.

(c) Bei langen Meetingsitzungen können die Energie-Kapazitäten von Mobilgeräten ausgeschöpft werden.

(d) Die Bandbreite ist meistens zu schmal, um Multimedianachrichten zu senden oder empfangen.

9. *Kabellose Büros*
Mitarbeiter, die zwischen verschiedenen Abteilungen abwandern müssen, haben die Möglichkeit, ihre kabellose Büros in Form von Notebooks mit Infrarot-Schnittstellen (z.B 802.11) mitzunehmen. Sie können damit überall (soweit die Reichweite einer Verbindung erlaubt) Emails senden und empfangen.
Noch ein Beispiel sind Anwälte, die ständig unterwegs sind und die Möglichkeit haben sollen, über ihre Post Bescheid zu wissen. Dazu brauchen Sie ein mobiles Endgerät und sichere, zuverlässige Funksysteme.

1.3 Gliederung

Diese Arbeit gliedert sich im einzelnen wie folgt auf: Im Anschluß an dieses **einführende Kapitel** erfolgt im **Kapitel 2** ein Überblick über den -Stand der Technik- vorhandene Messaging-Protokolle und -Architekturen und die damit verbundenen Standardisierungen (z.B X.500) und Mobilitätstechniken.
Daran anschließend folgt in **Kapitel 3** eine Beschreibung der Einsatztechniken von Messagingsystemen in einer mobilen Umgebung mit einem direkten Vergleich und Analyse im Bezug auf Mobilität.
Kapitel 4 beschäftigt sich mit einer aus Kapitel 3 resultierende Messaging-Architektur, die für eine mobile Umgebung geeignet ist. Diese wird auf ein Systemmodell abgebildet, und anschließend werden die einzelnen Komponenten, ihre Funktionalität und ihre Kommunikationsbeziehungen aufgezeigt.
Die daraus abgeleiteten Managementanforderungen werden in **Kapitel 5** entlang der Managementfunktionsbereiche aufgezeigt.
In **Kapitel 6** werden existierende Managementansätze für Messagingsysteme diskutiert, anschließend wird eine MIB spezifiziert und in **Kapitel 7** ein Implementierungskonzept dargestellt.
Kapitel 8 enthält eine Zusammenfassung und ein Ausblick.

1.4 Danksagung

Für Erstellung dieser Arbeit haben einige Personen beigetragen, bei denen ich mich an dieser Stelle bedanken möchte.
Mein besonderer Dank gilt meinen Betreuern, Herren *Stephen Heilbronner* und

Dr. René Wies und dem Lehrstuhl von Prof. Hegering in seiner Gesamtheit.
Ein spezielles Dankeschön geht auch an meine Korrekturleser, Herren *Richard Klings* und Frau *Julia*.

Kapitel 2

Standards und Architekturen-Stand der Technik-

In diesem Kapitel werden Standards und Architekturen von verschiedenen Messagingsystemen vorgestellet. Die Betrachtung aller existierenden Architekturen würde den Rahmen dieser Diplomarbeit überschreiten. Im folgenden wird auf Internet-Messaging besonders Wert gelegt.

2.1 Mailsysteme

2.1.1 Grundprinzipien

- **Funktionalität**
  Ein Mailsystem hat in der Regel zwei Hauptaufgaben:
  - Nachrichten an einen oder mehreren Empfnger zu erstellen und zu verschicken
  - Nachrichten zu empfangen und zu repräsentieren mit versciednen Funktionalitäten wie Replay- und Benachrichtungsmöglichkeiten

- **Komponenten eines Mailsystems**
  Die Benutzer interagieren mit einem Benutzeragent (UA) (laufender Prozess). Ein UA ist eine Werkzeugmenge, die wichtigtsten sind das zum lesen und das zum verschicken von Nachrichten. Der UA ist an einem Nachrichtenübertragungssystem (mehrere Transferagenten (MTAs)) angeschlossen, die MTAs sind Prozesse auf Rechnern, die über ein Netz verbunden sind. MTA's dienen der Übertragung von Nachrichten und Rückemldungen. Das Zusammenspiel zwischen UAs und MTAs sowie zwischen MTAs wird durch Protokolle festgelegt. Die wichtigste Komponente eines MTA's

ist die Routing-Tabelle, mit der Nachrichten weitergeleitet werden.
UA und MTA könnten durch einen Prozess realisiert werden.

- **Nachrichtenaufbau**
  Eine Nachricht ist einem Postbrief im Aufbau ähnlich. Sie besteht aus einem **Kopf**, der dem Briefumschlag entspricht und u.a: *Adresse, Datum, Ort, Zeitstempel, Carbon-Copy-Feld ...* enthält und einem **Rumpf** , der dem Briefinhalt entspricht, und kann: Text, Multimedia, ... sein.

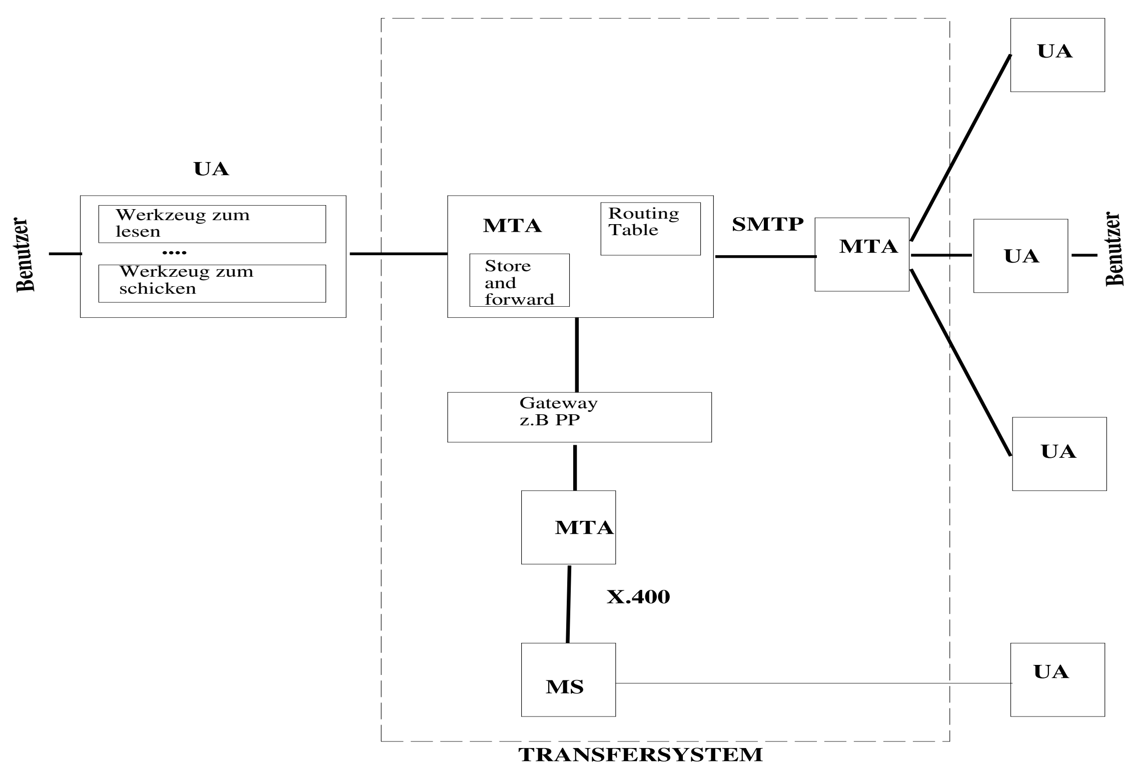


Abbildung 2.1: Komponenten eines Mailsystems

- **Gateways**:
  Diese werden zwischen unterschiedliche Mailsysteme geschaltet, sie konvertieren Nachrichten von einem Format in ein andreres.

2.1.2 Protokolle

Simple Mail Transport Protokoll: SMTP

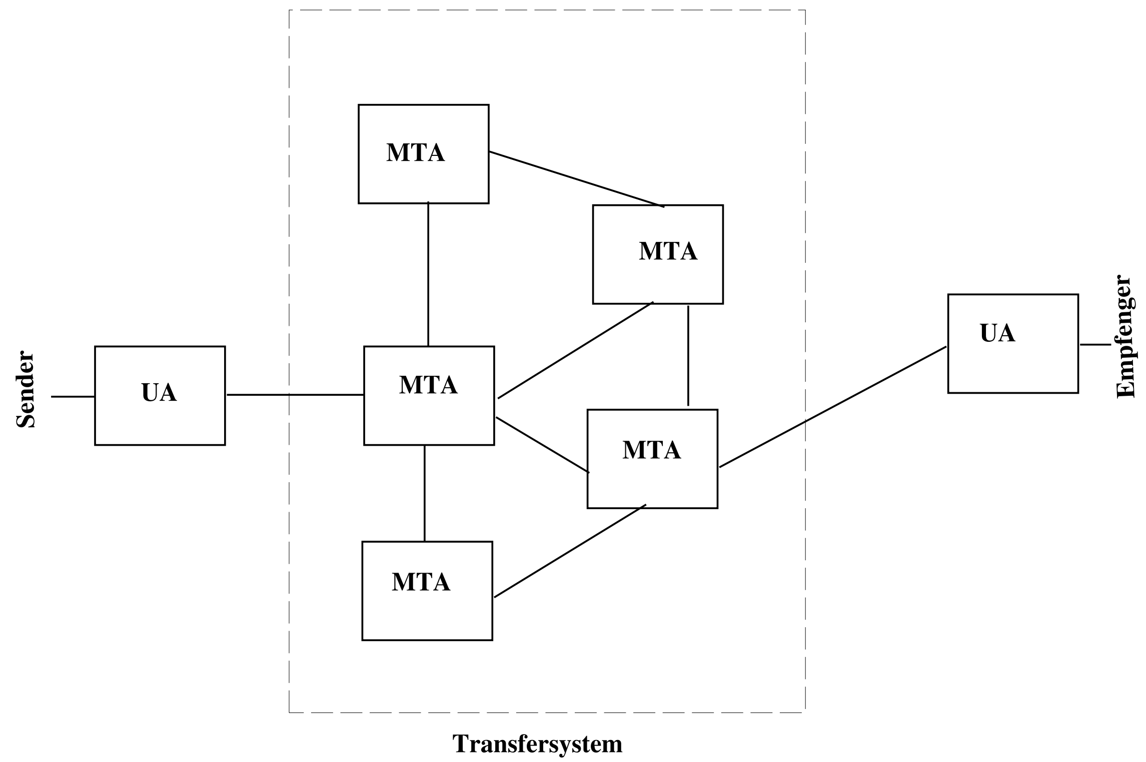


Abbildung 2.2: Architektur

Die Funktionalität von SMTP ([RFC821], [RFC822]) bezieht sich allein auf die Kommunikation von Textnachrichten. Eine Erweiterung um Multimediale Inhalte ist in MIME enthalten. Die Schnittstelle MTA-UA ist nicht genau definiert. Die Schnittstelle MTA-MTA ist festgelegt (SMTP). Die Kommunikation geschieht zwischen zwei Prozessen, der eine übernimmt die Rolle des Senders (üdermittelt Kommandos, die Parameter wie Adressen und Daten enthalten, zur Steurung der Nachrichtenübertragung) , der andere die Rolle des Empfängers (Schickt Rückmeldungen, die aus einen numerischen Code und eine Textuelle Erklärung bestehen). Die Nachrichten bestehen aus einem Kopf mit mehreren Felder wie to, subjekt, date, Cc und einem Rumpf, der eine Folge von Klartextzeichen enthält. Das wichtigste Feld des Kopfs ist die Adresse, die für das Routing wichtig ist. Die Adresse hat die Form name@ort wobei der Ort vom lokalen zum globalen aufgebaut ist.

SMTP Erweiterungen

SMTP wird ständig erweitert um die Anforderungen sowohl der Benutzer als auch der technischen Fortschritten zu erfüllen:

```
Date: Wed, 13 Dec 1995 09:40:56 -0500
From: Internet-Drafts@CNRI.Reston.VA.US
To: IETF-Announce:  ;
Cc: madman@INNOSOFT.COM
Subject: I-D ACTION:draft-ietf-madman-dsa-mib-1-00.txt
Parts/attachments:
   1   Shown   61 lines  Text
   2.1   OK   153 bytes  Message
   2.2 Shown   78 bytes  Message
----------------------------------------
```

Abbildung 2.3: Beispiel eines MIME-Nachrichtenkopfs

1. 8-Bit binäre Nachrichten ([RFC1426]), damit ist es z.b auch möglich Umlaute darzustellen.
2. Neue Nachrichten-Inhalte (siehe MIME).
3. Nachrichten Größe ([RFC1869]), ein SMTP-Server kann Nachrichten, die eine bestimmte Größe überschreiten ablehnen, das ist besonders wichtig für eine mobile Umgebung wegen der schmalen Bandbreite. Netzmanager können den maximalen Wert der Größe festlegen.

MIME

MIME ([RFC1521]) ist eine Erweiterung des RFC822 ([RFC822]), die das Nachrichtenformat festlegt.
Für die MTA's ist das Nachrichtenformat wie in RFC822 anzusehen, UA's müssen aber den Rumpf anders interpretieren. MIME unterstützt Multimedia-Mail, es wurden die Inhaltstypen: Text, Bild, Audio, Video, Nachricht (in der Nachricht), Anwendung (Binärdaten und Standard-Dokumente) und X-Token (Beliebige Inhalstypen mit Zwischenabsprache) definiert.
Mit MIME ist es möglich große Datenmengen zu referenzieren (*external-body*) und so wird es dem Empfänger überlassen, ob er die Daten holt oder darauf verzichtet. So kann auch eine Arbeitsgruppe an einem Objekt gemeinsam arbeiten. Die Dauer, wie lange die Daten gespeichert werden, bis sie abgeholt werden, ist in MIME nicht festgelegt. MIME erlaubt es ebenfalls Dateien an Nachrichten anzuhängen (siehe Abbildung 2.3).

MIME-Erweiterungen

1. Erweiterung um einen Neuen Inhalstyp ([RFC1894]) um Status-Report-Nachrichten austauschen zu können.

2. Erweiterung des Inhaltestypes Anwendung um sogenannte *White Pages Person Profile* ([HM96]).

3. MIME-Sicherheit mit Pretty Good Privacy ([Elk95]).

X.400

X.400 ([X.400]) ist ein CCITT-Mail-Standard. X.400 ist ähnlich wie SMTP in der Architektur und Funktionalität, ist aber wesentlich komplizierter und umfangreicher. Es unterscheidet sich von SMTP in :

- Der Message Store: Das ist eine zusätzliche Komponente, die zwischen UA und MTA eingeschaltet werden kann, so daß Nachrichten für eventuell ausgeschaltete UA's nicht verlorem gehen.

- Zusätzliche Dienste wie das Schicken von Test- oder Bestätigungs-Nachrichten.

- Die Inhaltstypen einer Nachricht: neben Text auch Fax, Telefax und Sprache.

- Das Adressverzeichnis X.500: Siehe Abschnitt 2.3.

- Die Kommunikation zwischen den Komponeneten geschieht gemäß verschiedener Standards wie in Abbildung 2.4 dargestellt ist.

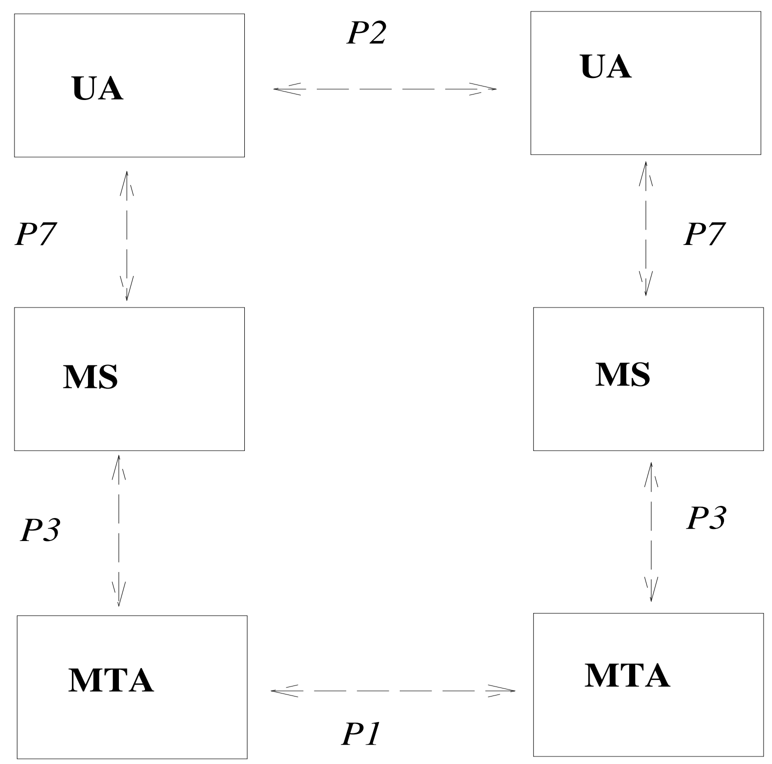


Abbildung 2.4: X.400-Architektur

Post Office Protokoll: POP-3

POP-3 ([RFC1460]) ist ein Mail-Abhol-Protokoll, mit dem Mails abgeholt aber nicht gesendet werden können.
POP-3 hat eine Klient-Server Architektur und ist besonders für PC's, die nicht immer eingeschaltet sind entwicklet worden. Ankommende Nachrichten werden in einem Server (POP-3-Server) gespeichert und auf Wunsch dem Klient weitergeleitet. Jedem User Agent ist **eine** Mailbox zugeordnet. Die Nachrichten werden von eins bis zur momentanen Anzahl der Nachrichten numeriert.

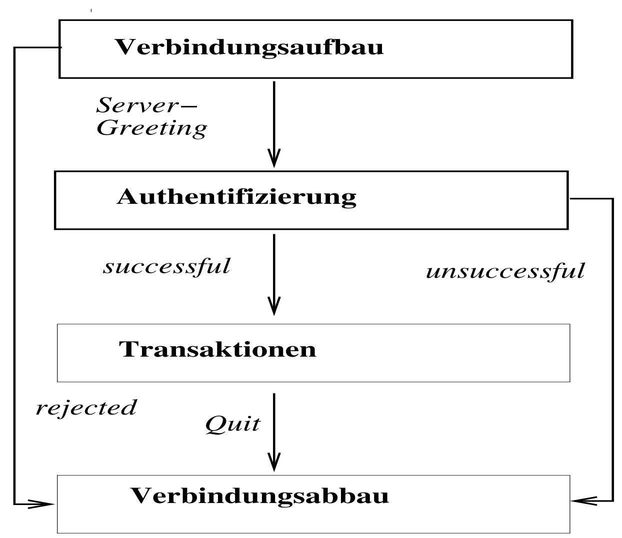


Abbildung 2.5: Eine POP-3-Sitzung

Der User Agent (Klient) baut eine Verbindung zum Server auf. Der Server antwortet mit einem Greeting. Nach der Authentifizierung, diese ist entweder mit `LOGIN/PASSWORT` oder mit `NAME` und der Message Digest Algorithmus (MD5). Nach der Authentifizierung fänget die Sitzung an vobei der Klient Kommandos sendet, die vom Server beantwortet werden. Eine Sitzung endet bei nicht Authentifizierung, mit dem `QUIT`-Kommando oder Server-Schutdown.

Eigenschaften:

- Architektur : Klient-Server
- Authentifizierung: Name/MD5, in Erweiterungen ([RFC1734]) auch mit externe Sicherheitsmechanismen (KERBEROS).
- Nachrichten:
  - Wahlweise Laden oder löschen
  - Teile und Köpfe Laden
  - Grösse und Anzahl abfragen
- Abfragen, welche Nachricht zuletzt bearbeitet wurde

Interactive Mail Access Protokoll: IMAP-4

IMAP-4 ([RFC1730], [RFC1731], [RFC1732], [RFC1733] ist auch ein Mail-Abhol-Protokoll. Dieses Protokoll legt fest wie ein Mail-Klient mit dem Server kommuniziert. IMAP-4 unterstützt multiple Mailboxen, d.h. ein User Agent kann eine oder mehrere Mailboxen haben. Der Klient kann die Mailboxen und die Nachrichten remote verwalten und manipulieren als ob sie lokal wären.

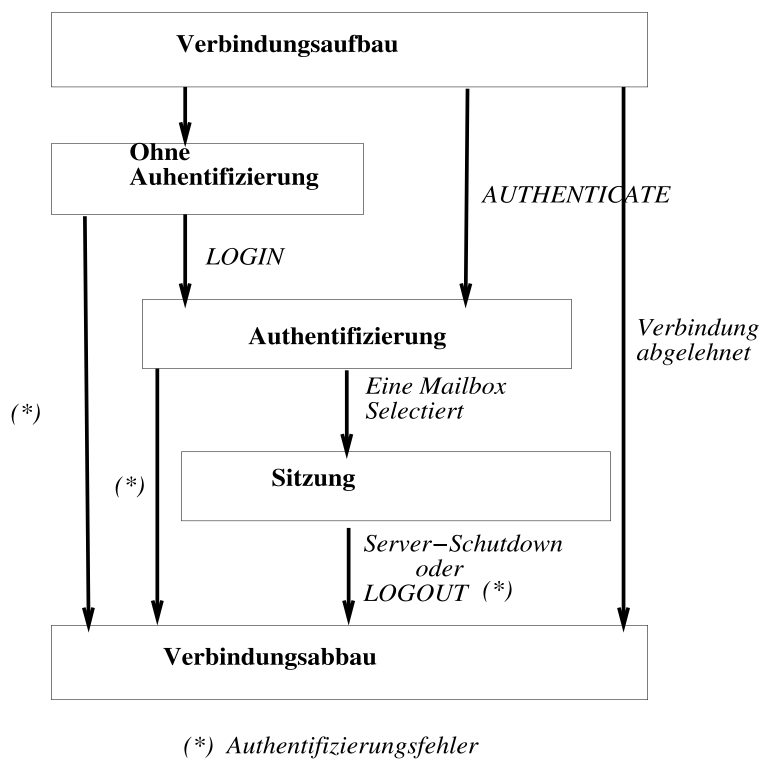


Abbildung 2.6: Ablauf einer IMAP4-Sitzung

Eine Gruppe von Benutzern kann auch eine gemeinsame Mailbox haben. Der Zugriff auf Mailboxen kann entweder zum Lesen, zum Lesen und Schreiben oder zum Lesen und Zufügen (News-Groups) sein.
Nachrichten in einer Mailbox haben Sequenznummern oder Identifikatoren zur eindeutigen Identifizierung und zur Synchronisation zwischen Klient und Server.
Die aktuelle IMAP-4-Version unterstützt nur **einen Server** und ist nicht zum Senden konzipiert wurden. Eine wichtige Eigenschaft von IMAP-4 ist, daß es nicht nur RFC822 (POP-3) unterstützt, sondern auch **MIME** .
Eine IMAP-4-Sitzung beginnt mit einem Server Greeting (Siehe Abbildung 2.6). Danach tritt die Authentifizierungsphase, diese könnte durch `LOGIN/PASSWORT` oder ein externes Sicherheitsmechanismus (z.B KERBEROS), das dann für die ganze Sitzung gilt, geschehen. Der Klient selektiert dann eine Mailbox, um Nachrichten, Nachrichten-Flags oder Mailboxen zu bearbeiten.
Die Mailbox-Namen müssen hierarchisch (Links-nach-Rechts) aufgebaut, der Name `INBOX` ist für jeden Benutzer reserviert. Das Interpretieren der Mailbox-Namen ist sonst implementierungsabhängig.
Eigenschaften:

- Mailboxen:

  - Die Wahl einer beliebigen Mailbox, um die darin enthaltenen Nachrichten zu bearbeiten oder **nur** zu lesen.
  - Kreieren, löschen, umbenennen, registrieren, listen,... von Mailboxen.
  - Nachrichten in einer Mailbox **zufügen**

- Suchen in einer Mailbox nach bestimmten Kriterien. Diese können standard oder benutzerdefiniert sein, wie z.B. Nachrichten-Flags (z.B: `new, answered, draft, larger,...`).

- Nachrichten:

  - Teile einer Nachricht, Kopf einer Nachricht oder die ganze Nachricht nach bestimmten Kriterien laden. Besonders wichtig ist die Möglichkeit MIME-Multipart-Nachrichten zu bearbeiten.
  - Größe einer Nachricht oder eines Teiles abfragen.
  - Nachricht von einer Mailbox in einer anderen kopieren.
  - Selbsteditierte Nachrichten in einer Mailbox zufügen.

- Möglichkeit der Erweiterung der Klient-Kommandos.

- *IMAP-4 hat angesichts der Nachrichten-Flags (Standard oder Benutzerdefiniert) und MIME-Parsing große Vorteile, die im nächsten Kapitel erläutert werden.*

2.2 Paging-Netze

2.2.1 Einführung

Pagingdienste unterscheiden sich in der Art des Dienstes und des Einsatzes.

- **Art**
  Voice-Paging (Sprache), Tone-Paging (Töne), Display-Paging (Numerisch, Alphanumerisch).

- **Einsatz**

  - Warnsysteme: z.B. Feuerwehr, Verkehr.
  - Finanzinfomationen : z.B. Börsennachrichten.
  - Wetter, Uhrzeit, Fluginformationen, Bahninformationen.
  - Industrie : Produktionsmaschinen.

- **Europäische Dienste**

- **Cityruf**
  Ton, numerisch (15 Zeichen), alphanumerisch (80 Zeichen).
  Regionale Bereich (Städtenahe Zonen).
  Abrechnung: Anzahl der gebuchten Zonen, Einzel oder Gruppenruf, Rufklasse.
- **Eurosignal**
  Einzel, Gruppenruf.
  Alle Rufklassen (verschiedene Signale), Bedeutung der Signale muß allerdings zwischen den Partnern vereinbart werden.
- **European Radio Messaging System**
  Bietet auch Datenübertragung.
  24 Kanäle im Frequenzband zwischen 169,9 MHz und 169,8 MHz
  Benutzung eines Roaming-Pager zur Suche des Frequenzbandes nach freien Kanälen, Vorteil ist das Dienstanbieter nicht alle Kanäle zur Verfügung stellen müssen .

2.2.2 Simple Network Paging Protokoll: SNPP

Motivation

SMTP ist zuverlässig, Nachrichten kommen aber nicht immer rechtzeitig. Der Grund dafür ist die *Store-and-Forward* Technik sowohl bei SMTP als auch bei IP. Nachrichten können zwar mit SMTP sicher ankommen, das kann aber sehr lange dauern durch den obengenannten Grund sowie die wiederholten Zustellversuche im Fall des Ausfalls eines MTA's oder Gateway (Nachrichten werden nach einer bestimmte Zeit wieder zugestellt). Aus diesen Gründen ist SMTP für Eilnachrichten nicht geeignet.

Architektur

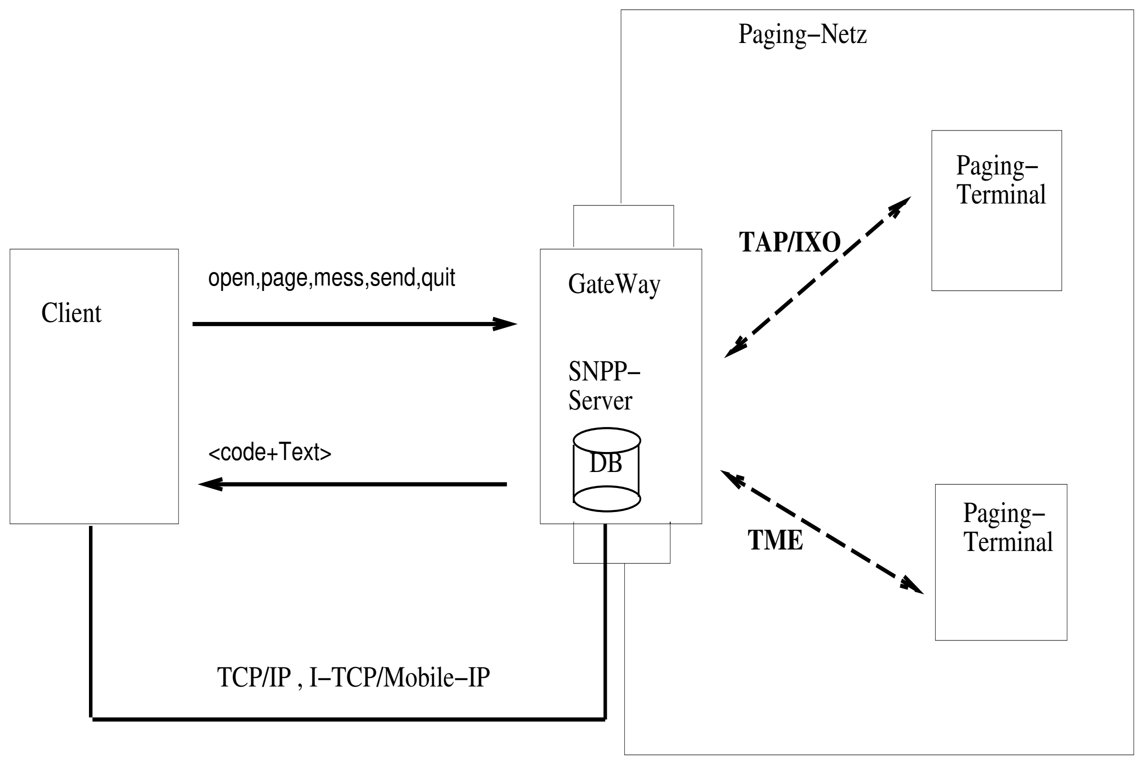


Abbildung 2.7: SNPP-Architektur

SNPP ([RFC1568]) schafft eine direkte Verbindung zum Paging-Server (gilt als Gateway zum Paging-Netz) ohne zusätzliche SMTP-Relays (MTA's Store-and-Forward), die Antworten vom Server folgen erst nach der Abfrage des Pagers. Eine positive Antwort heißt, daß die Nachricht tatsächlich zum Paging-Netz geliefert ist. Dadurch wird eine Benachrichtigung des Benutzers direkt möglich. SNPP unterstützt numerische und alphanumerische Nachrichten, eine Nachricht kann nur an **einen** Empfänger geschickt werden. Die Aufgabe vom Server ist es unter anderem, Nachrichten mit Hilfe einer Datenbank zu formatieren und zum TAP (Telecator Alphanumeric Protocol) Terminal zu schicken, dann auf eine Reply warten, die dem Klient übermittelt wird. Die Kommunikation geschieht mit dem TAP Terminal geschieht mittels TAP/IXO, der die Gültigkeit von PID's nur wenn die Nachricht von Paging-Terminal akzeptiert wird, erkennt.
Zusammengefaßt heißt SNPP : **Nachricht direkt und sofort schicken oder Fehler melden**.

Erweiterungen von SNPP

Die spätere Versionen von SNPP enthalten Erweiterungen, die im Zusammenhang mit TME (Telocator Message Entry Protocol) entwickelt wurden.

- **SNPP: Level2 ([RFC1645])**
  - Sicherheitsmechanismus : Login und/oder Paßwort.
  - Senden von mehr als einer Zeile (wie SMTP : DATA-Kommando).
  - Verschiedene Service-Level.
  - Alarmieren von Empfänger bevor sie einen Nachricht bekommen.
  - Rufzonen überschreiben (z.B München statt Deutschland) oder Pagingdienst ändern.
  - Senden mit Zeitverzögerung.
  - Nachrichten-Subject angeben (wie SMTP).
- **SNPP: Level3 ([RFC1730])**
  Two-Way-Paging.

2.3 Adressverzeichnisse: X.500

Aufgaben eines Mail-Systems sind im wesentlichen : Nachrichtentransfer, Konvertierung von Nachrichtenformaten, und Adressen und die Synchronisation der Adressverzeichnisse, wobei die letzte Aufgabe eine große Schwierigkeit in einer heterogenen Umgebung darstellt. Der X.500 ([X.500]) Verzeichnisdienst bietet eine Lösung für dieses Problem.

- X.500 ist ein globales Verzeichnis
- X.500 ist hierarchisch strukturiert, vom lokalen zum globalen.
- X.500 enthält sowohl Benutzer als auch Ressourcen.
- X.500 ist eine Menge von Objekten, jedes Objekt hat dann eine Menge von Attributen z.B : Das Objekt Person könnte die Attribute Name, Adressen (Verschiedene Email-Systeme), Nachrichten-Format Information und Routing-Information haben.
- Dieses Verzeichnis hat eine Baumstruktur. Der Directory System Agent (DSA) ist zuständig für die Einträge auf einer bestimmten Tiefe im Verzeichnisbaum (siehe Abbildung), der Directory User Agent (DUA) dient der Kommunikation zwischen einem Benutzer oder einem Benutzerprozeß und dem DSA, um Information aus dem Verzeichnis zu holen. Der DUA kommuniziert nur mit einem DSA, diese kommuniziert mit den anderen DSAs um die gewünschte Information herauszuholen. DUA und DSA nutzen das Directory Access Protokoll (DAP), um Informationen auszutauschen.

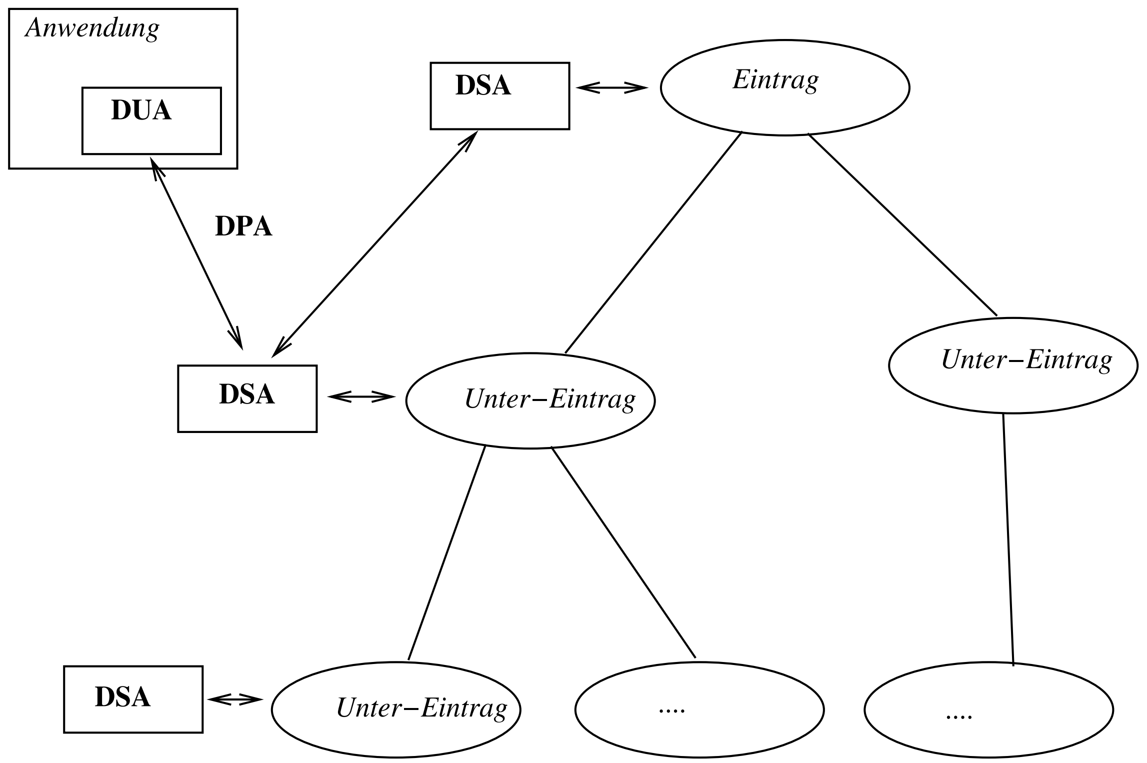


Abbildung 2.8: Heterogene Umgebung mit dem X.500 Verzeichnis

2.4 Externe Sicherheitsmechanismen

2.4.1 Einleitung

Grundbegriffe

- **Authentifizierungscode**
  Erkennung von zufälligen und gezielten Veränderungen von Nachrichten. Sei M eine Nachricht, M wird um spezielle redundante Information ergänzt, die mittels eines kryptografischen Verfahrens aus M berechnet und zusammen mit der Nachricht gespeichert bzw. übertragen werden. Für einen Abhörer ist es damit unmöglich, eine zweite Nachricht mit der gleichen kryptografischen Prüfsumme zu konstruieren. Die Ermittlung der redundanten Information geschieht unter Verwendung eines geheimen Schlüssels (K). Dieser Vorgang wird MAC (Message Authentication Code) genannt.

Authentifizierungsmechanismen

- **Paßwort oder Personal Identification Number (PIN)**

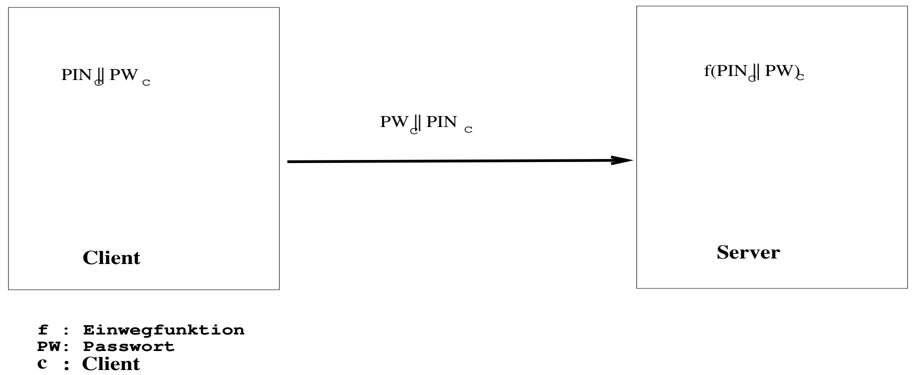
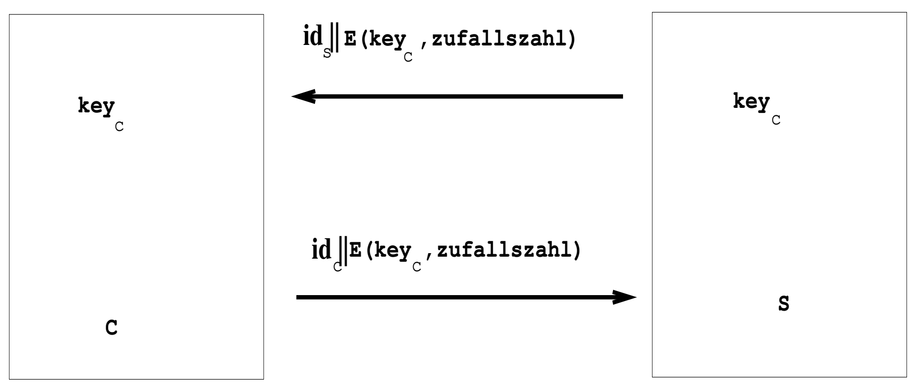


Abbildung 2.9: Authentifizierung durch PIN oder Passwort

Die Paßwörter sind nicht im Klartext, sondern mit Hilfe einer Einwegfunktion f gespeichert [FR94]. Der Server berechnet $f(PW_c)$ oder $f(PIN_c)$ und vergleicht das Resultat mit dem zu PW_c bzw. PIN_c gehörenden Wert. Ein Problem besteht aber in dem Fall, wenn es kein sicheren Kanal zwischen Klient und Server gibt. Ein Abhörer könnte sich **als Klient identifizieren** und auch bei verschlüsselten Authentifizierungsdaten besteht die Gefahr, daß diese Daten **wiedereingespielt werden**.

- **Einsatz von Zeitstempeln**
  Die Nachrichten werden mit Zeitstempeln versehen. Das hat zur Folge, daß die oben genannten Angriffsmöglichkeiten vermieden werden können. Dabei entstehen aber neue Probleme und Forderungen :
  - Die Systemzeit muß exakt und zuverlässig sein.
  - Die Synchronisation der Uhren der beteiligten Kommunikationspartner stellt eine Schwierigkeit dar.
  - Eine Nachricht wird akzeptiert, falls die Abweichung zwischen dem Zeitpunkt des Nachrichteneingangs und Zeitstempel einen Schwellwert nicht übersteigt. Eine Wiedereinspielung innerhalb dieser tolerierten Zeitintervalls ist möglich, die Lösung wäre eine Buchführung über akzeptierten Nachrichten. Problematisch ist die Übertragungsverzögerungen zu erkennen, sowie Nachrichten vor Manipulation zu schützen.
- **Einsatz von Sequenznummern (logische Zeitstempel)**
  Zu jeder Sequenznummer wird nur eine Nachricht und in dem Fall das dieser Sequenznummer größer als die entsprechende gespeicherte Nummer ist, akzeptiert. Ein Numerierungsverfahren muß von den beteiligten Kommunikationspartnern initiiert werden. Dieses Verfahren ist zwar einfach in Protokollen zu integrieren hat aber auch Nachteile wie Synchronisationsprobleme im Fall eines Systemausfalls und einen großen Verwaltungsaufwand.

- **Challange und Response**



E : Chiffrierung.
c : Client.
id: Identit"at.
S : Server

Abbildung 2.10: Challange and Response

Die verifizierende Stelle übermittelt der sich authentifizierenden Stelle eine Zufallszahl (z.B Pseudo-Zufallszahl), die als challange bezeichnet ist, und fordert eine Antwort (Response) in einem bestimmten Zeitintervall (timeout). Das Vorteil ist ein geringer Verwaltungsaufwand, durch die zusätzliche Transaktion wird aber der Netz belastet. Ein Reflexionsangriff ist auch nicht auszuschließen.

Schlüsselmanagement

- **Allgemeine Anforderungen**
  - Geringer Aufwand im laufenden Betrieb, d.h. insbesondere die Vermeidung manuelle Prozeduren und aufwendiger Initialisierungsprozesse.
  - Möglichst geringe Anforderungen an physische Sicherheitsmaßnahmen (z.B Sichere Kanäle).
  - Flexibilität bezüglich der verwendeten Schlüsselverteilmechanismen und damit Anpaßbarkeit an neuen Kenntnissen.
  - Eine möglichst geringe Anzahl und Komplexität vertrauenswürdigen Mechanismen, insbesondere die Vermeidung einer zu großen Zentralisierung sensitiver Information.

- **Schlüsselerzeugung**
  Die ausgewählte Methode hängt von verschiedenen Kriterien ab. Die wichtigsten sind die Lebensdauer der Schlüssel, der Verwendungszweck, Art der Verschlüsselung (Daten, Kommunikation, Schlüssel selber), da die Lebensdauer des Schlüssels davon abhängt. Schlüssel müssen von einer vertrauenswürdigen Stelle in einer Umgebung, die gegen Ausfälle gesichert ist, erzeugt. Die Schlüsselerzeugung soll möglichst mit echten Zufallsprozessen z.B radioaktiven Zerfall oder Widerstandsrauschen geschehen (benutzergewählte Paßwörte und Schlüssel stellen eine Gefahr für das System). Häufig werden Pseudozufallsgeneratoren (deterministischer Algorithmus) eingesetzt.

2.4.2 Privacy Enhanced Mail (PEM)

PEM ([Zeg93]) ist ein Mechanismus, der für die Sicherheit in einem Emailsystem konzipiert wurde. Ein User Agent soll mit einer PEM-Software ausgestattet werden um PEM gesicherte Nachrichten versenden und empfangen zu können.
Jeder User Agent mit PEM-Software verfügt über einen Eintrag genannt *Zertifikat* in einem öffentlichen Verzeichnis (z.B X.500). Die Zertifikate enthalten Informationen über den Benutzer z.B der Benutzer öffentliche Schlüssel, mit dem Daten verschlüsselt werden können, die diesem Benutzer zugesendet werden. Neue Benutzer können sich im Verzeichnis durch das Certification Authority, das für das Management von Zertifikate Zuständig ist, registrieren. Die benutzer-eigene Informationen wie der Privat-Schlüssel, mit dem der Benutzer in der Lage ist Nachrichten, die mit seinem öffentlichen Schlüssel verschlüsselt worden sind, zu entschlüsseln, werden in einem sogenannten *Personal Secure Environment* (PSE) gespeichert.
Die Benutzer generieren ihren Nachrichtenrumpf mit Hilfe der PEM-Software (Die Nachricht wird verschlüsselt) und schicken sie mit dem user Agent. Die Kommunikation geschieht wie üblich mit SMTP. Der Empfänger entschlüsselt dann die Nachricht mit Hilfe seiner PSE.
Die PEM-Software holt die Zertifikate und überprüft ihre Gültigkeit mit Hilfe der *Certification Revocation List* (CRLs) automatisch.
PEM bietet verschiedene Sicherheitsservices: Die Integrität der Daten während einer Sitzung, die eindeutige Authentifizierung des Senders, die Überprüfung der Gültigkeit des Senders und der Herkunft (Origin) der Daten.
Sie deckt aber nicht alle Sicherheitsaspekte, so bleiben die eindeutige Identifizierung des Empfängers und die Zugriffskontrolle auf die Ressourcen unberücksichtigt.

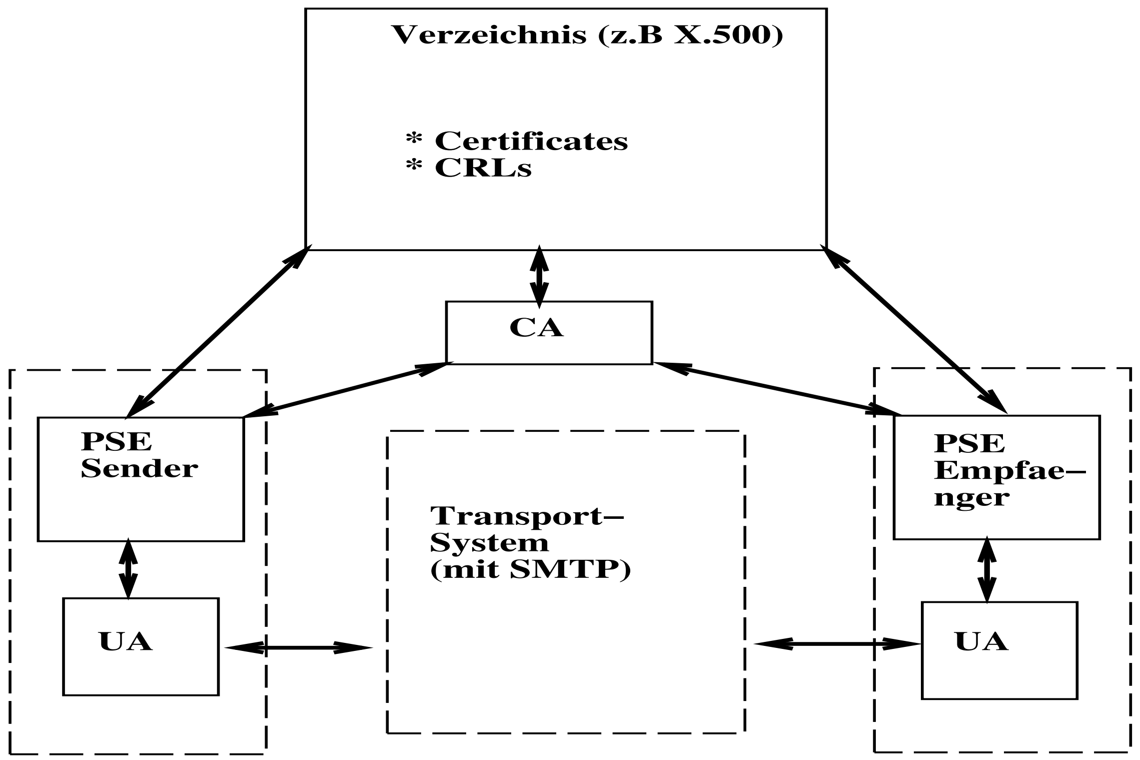


Abbildung 2.11: PEM-Architektur

PEM benutzt das RSA-Verfahren, RSA ist wegen der langen Schlüsseln unbeliebt, die Rechenzeit für die Ver- und Entschlüsselung große Datenmengen ist sehr aufwendig

2.4.3 KERBEROS V4

KERBEROS V4 ([RFC1510]) ist für Klient-Server-Anwendungen konzipiert und eignet sich besonders für physikalische und sonst ungeschützte Transportverbindungen. Diese Version von Kerberos arbeitet nur mit TCP/IP-basierten Netzen. Eine Implementierung von Kerberos besteht in der Regel von einem Authentifizierungszentrum (Key Distribution Center: KDC), das physich geschützt ist und eine Subroutinen-Bibliothek , die von Anwendungen benutzt wird um ihren Benutzern zu identifizieren.
Eine Klient-Server-Sitzung kann mit Kerberos wie folgendes beschrieben:
Der Klient sendet einen Request an den authentifizierenden Server (KDC), der dem Klient dann ein *Ticket* und ein Sitzungsschlüssel verschlüsselt in dem Klient-Privatschlüssel übermittelt. Das Ticket enthält den Klient-Name und eine Kopie vom Sitzungsschlüssel verschlüsselt im Server-Privatschlüssel. Der Klient sendet das Ticket für den Server weiter. Der Sitzungsschlüssel ist damit beiden Kommunikationspartnern bekannt und so kann eine Sitzung beginnen.

Der Authentifizierungsserver pflegt eine Datenbank, die Informationen über Klients und Server wie die Privatschlüssel enthält, dieser Server soll physisch gesichert werden.
Klient und Server verfügen über Code-Bibliotheken um Daten zu verschlüsseln und zu entschlüsseln. Beide Kommunikationspartner können auch den von Authentifizierungsserver zugeteilte Sitzungsschlüssel um einen neuen Sitzungsschlüssel zu vereinbaren.
Replay-Angriffe werden dadurch vermieden, daß ein Zeitstempel mit dem Ticket zugesendet wird.
Damit sich die Kommunikationspartner gegenseitig authentifizieren, werden Identifikatoren, die vom Authentifizierungsserver mit den Privatschlüsseln der beiden verschlüsselt werden, verwendet. Die Integrität und Vertraulichkeit der Daten werden gegen Angriffe wie z.B die Manipulation des Datenstroms mit dem Sitzungsschlüssel sowie eine Prüfsumme geschützt.

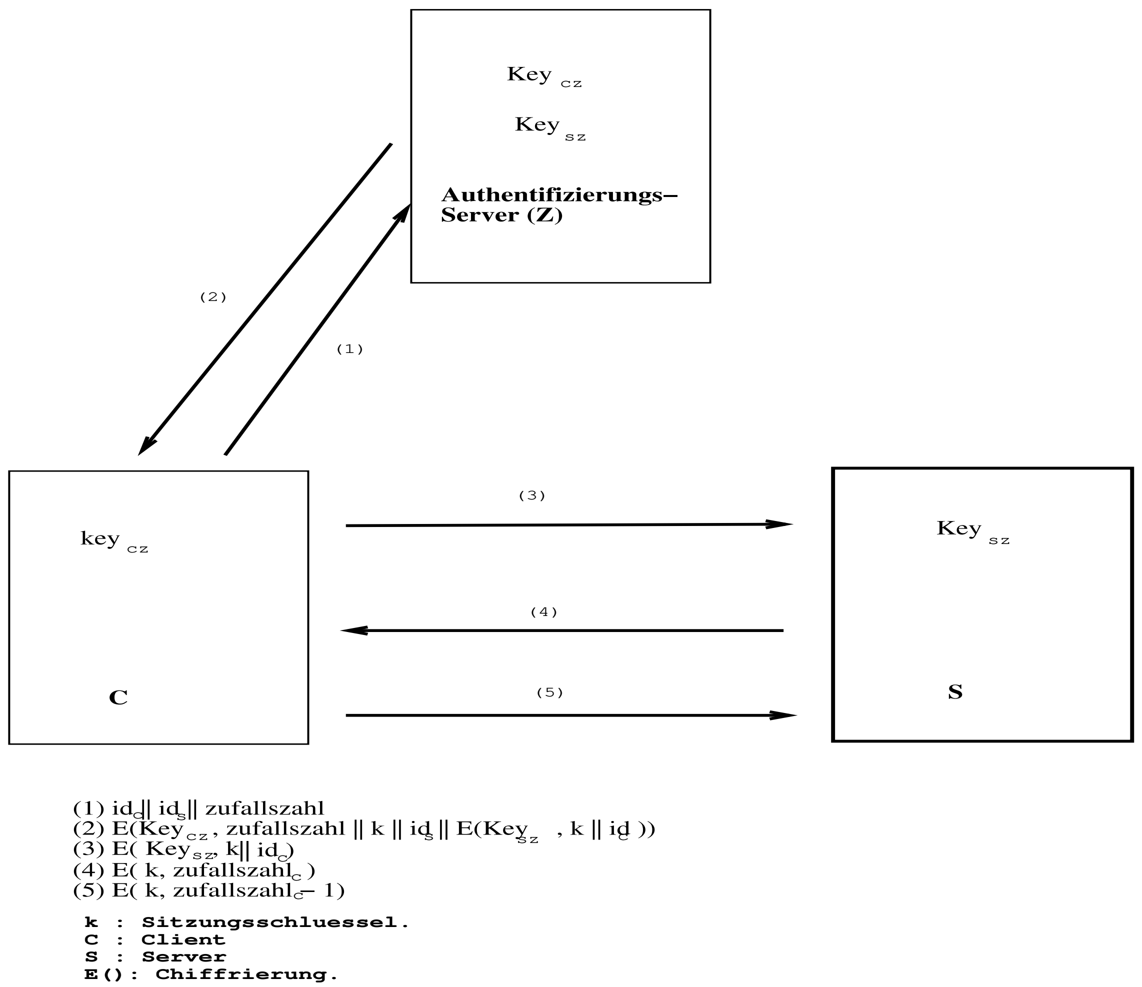


Abbildung 2.12: Beispiel einer Kerberos-Implementierung mit Zufallszahl statt Zeitstempel

KERBEROS basiert zwar auf einem sicheren aber komplizierten kryptographischen Verfahren.

2.4.4 Pretty Good Privacy (PGP)

PGP ([KPS95]) wird auch mit IMAP-4 eingesetzt. Eine Nachricht wird zuerst mit dem privaten Schlüssel des Absenders verschlüsselt, damit wird die Authentität des Absenders gesichert und anschließend dem öffentlichen Schlüssel des Absenders verschlüsselt. Dieser entschlüsselt die Nachricht mit seinem privaten Schlüssel und dann mit dem öffentlichen Schlüssel des Absenders.
PGP generiert für jede Verschlüsselung einen *One-Way*-Schlüssel, der mit der Nachricht verschlüsselt und verschickt wird (*Ein PGP-Schlüssel zu knacken ist vergleichbar mit einem RSA-Schlüssel mit 3100 bit zu faktorisieren*). PGP speichert die Privaten und öffentlichen Schlüssel getrennt. Jeder Schlüssel hat eine

Kennung bestehend aus 64 bit, die von PGP intern benutzt werden.
PGP benutzt den MD5-Algorithmus um aus eine Nachricht einen 128-bit-Zahl zu generieren, diese gilt als Unterschrift für die Nachricht (Prüfsumme) damit wird garantiert, daß eine Nachricht nicht verändern werden könnte.
Replay-Angriffe werden in PGP mit Zeitstempeln verhindert. Für die Erzeugung der obengenannten *One-Way*–Schlüssel wird ein Pseudo-Zufalls-Generator verwendet, der Startwert wird aus dem zeitlichen Abstand von Tastatureingaben berechnet.
Die Berechnung des Algorithmus geschieht dann mittels des Verschlüsselungs-Algorithmus IDEA, der 128-bit langen Schlüsseln verwendet. PGP bietet auch ein Datenkomprimierungstool an.
PGP braucht kein sicherer Kanal zum Schlüsselaustausch, ist aber ursprünglich für Einzelplatzrechner gedacht.

2.5 Mobility-Techniken

2.5.1 Mobile-IP

Mobile-IP ([Per96], [HN95]) ermöglicht ein Routing der IP-Datagrammen zu mobilen Knoten im Internet. Die Mobilstation verfügt über zwei IP-Adressen: eine auf dem heimatlichen Subnetz, unter der sie stets erreichbar ist, und eine für ihre Aufenthalte in einem fremden Netz genannt *care-of-adress*, diese ist entweder die ursprüngliche Adresse, oder ein sogenannter *Mobile-Agent*, falls Mobile-IP auf diese Station nicht implementiert ist.
Ein Paket wird zuerst an die Heimat-Adresse geschickt, dort wird er geroutet von *Home-Agent* zur *care-of-adress* falls die Mobile-Station sich in einem fremden Netz befindet. IP wird ständig erweitert (IPng, IPv6), um Probleme wie dem ungenügenden Adressraum zu vermeiden.

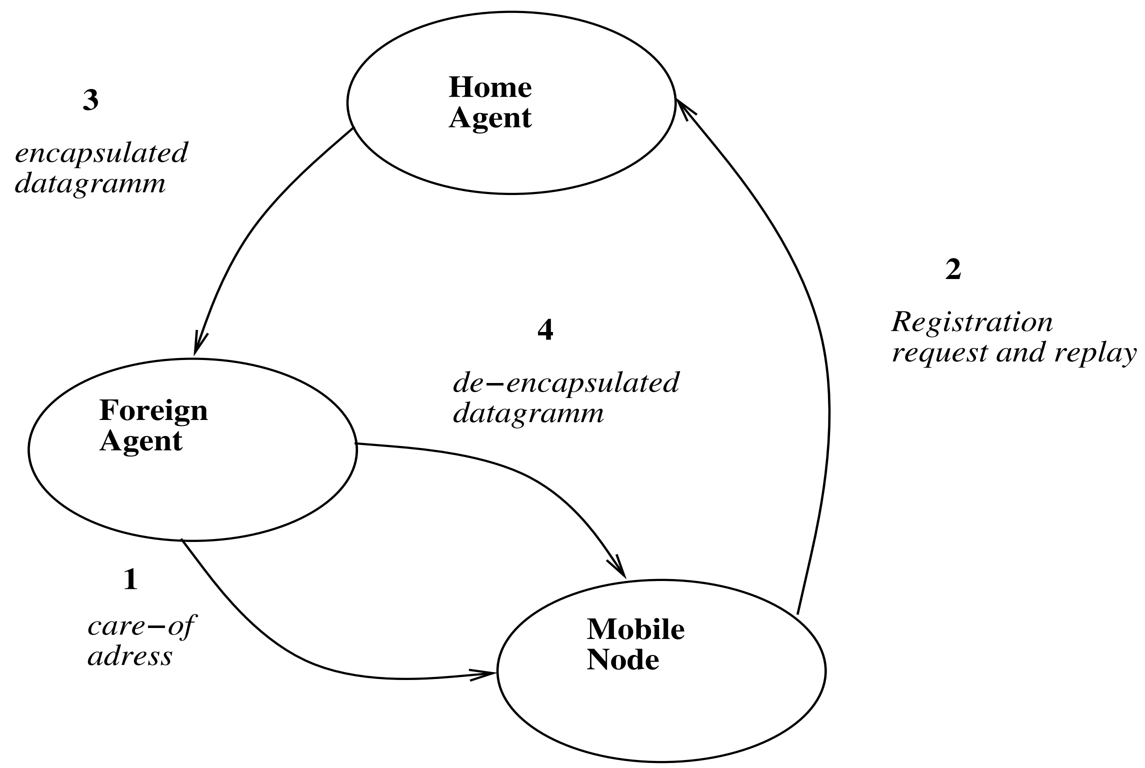


Abbildung 2.13: Mobile-IP

2.5.2 802.11

802.11 ([Wei95]) ist ein IEEE-Standard, der für drahtlose Netze entwickelt wurde. 802.11 ist ein verteiltes Medium-Zugriffsprotokoll auf dem andere Dienste wie Strommanagement und Synchronisation aufbauen. Ein drahtlose Netz nach 802.11 besteht aus mindestens zwei Stationen (*Basic Service set*) und ein *Portal* zur Anbindung an anderen Netzen. Mehrere Stationen sind mit einem *Distribution System* verbunden. Stationen werden durch eine *Distributed Coordination Function* (DCF) gesteuert.
Die DCF ist unabhängig von der physikalischen Schicht. Die Sender-Station überprüft, ob das Übertragungsmedium für eine gewisse Zeit frei ist. Wenn nicht, wartet die Station eine zufällig gewählte Verzögerungszeit. Der Test, ob der Übertragungsmedium frei ist, erfolgt auf physikalischer Ebene oder als virtueller Mechanismus. Es werden spezielle kurze *Frames* übertragen, um Kollisionen zu vermeiden und anderen Stationen mitzuteilen, wie lange bei einer Übertragung das Medium belegt ist. Dieses Mechanismus kann auch durch sogenannten stationseigenen Steuerparameter ersetzt werden, um große Overheads (verursacht durch die kleinen *Frames*) zu vermeiden . Für jede Station wird ein *Contention Window (CW)* der zwischen CW_{min} und CW_{max} variiert, festgelegt. Mit jedem neuen Übertragungsversuch verdoppelt sich der CW. CW geht wieder auf CW_{min}

wenn die Übertragung erfolgreich ist. Wenn ein Medium belegt ist, werden alle wartende Stationen für eine festgelegte Zeit (DIFS) in eine Schlange gesetzt. Eine Station, die senden will, berechnet zuerst eine zufällige Startzeit in CW ($= CW * Random * Slot - Zeit$).
Nach dem DIFS wird versucht zu senden. Falls das Medium belegt ist, läuft dann der Timer, bei 0 wird der Frame gesendet. Die Station soll dann für jeden Frame eine zufällige Verzögerungszeit auswählen.

2.5.3 Cellular HLR/VLR am Beispiel GSM

Funkkanäle

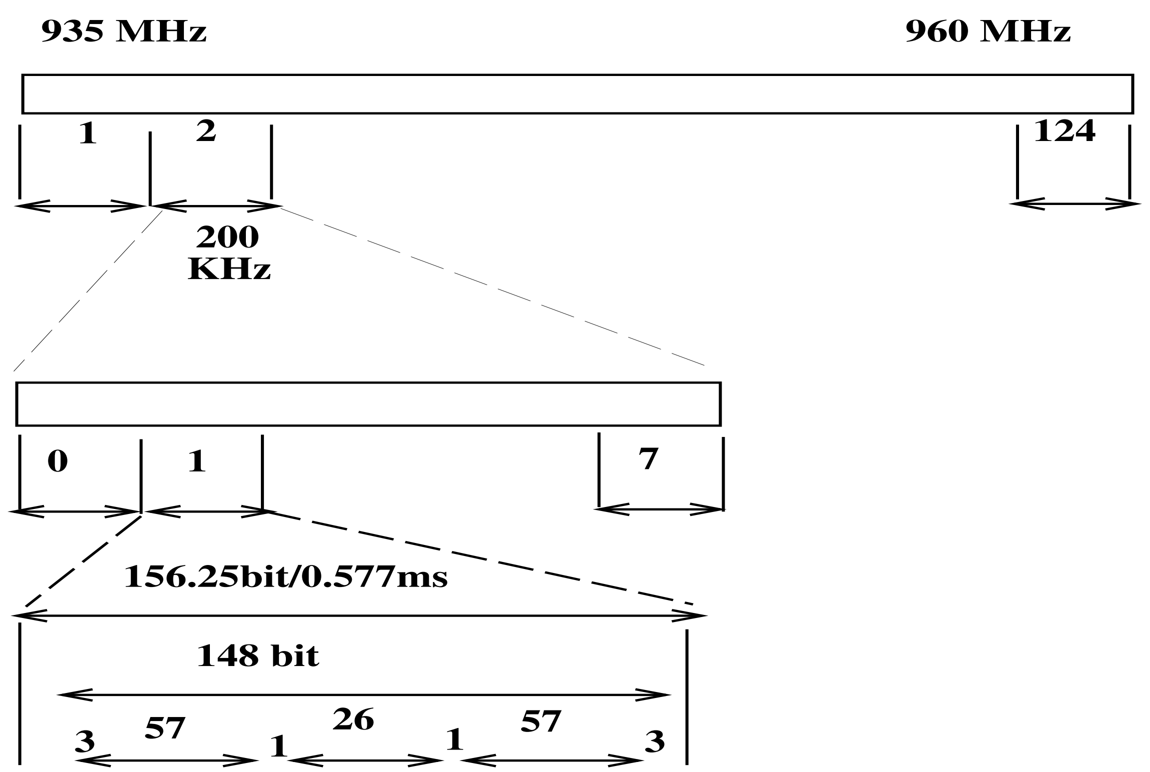


Abbildung 2.14: TDMA bei GSM

Mobilen Netzen werden in der Regel Frequenzbereiche zugeteilt, GSM[1] ([MP91]) ist zum Beispiel der Frequenzbereich von 890-915 MHz (Signale von einer Mobilstation zur Basisstation) und 935-960 MHz (Kommunikation Basisstation und Mobilstation) zugeteilt. Pro GSM-Funkzelle stehen dann 124 Kanalpaare (siehe Abbildung 2.14) mit je 200 KHz Bandbreite. Die Übertragung in einem GSM-Netz basiert auf dem *Time Division Multiple Access* (TDMA), dabei wird jeder Kanal in acht *Slots* geteilt, jeder ist 0.577 Millisekunden lang, in dieser Zeit

[1] HLR: Home Location Register, VLR: Visitor Location Register

können 156.25 Bits übertragen werden. Unter Berücksichtigung der Sicherheitsabständen bleiben 148 bit für ein Slot. Dieser wird in zweimal 57 Datenbits und 26 Kontrollbits zur Synchronisation und messen der Kanalqualität aufgeteilt.

Management des GSM-Netzes

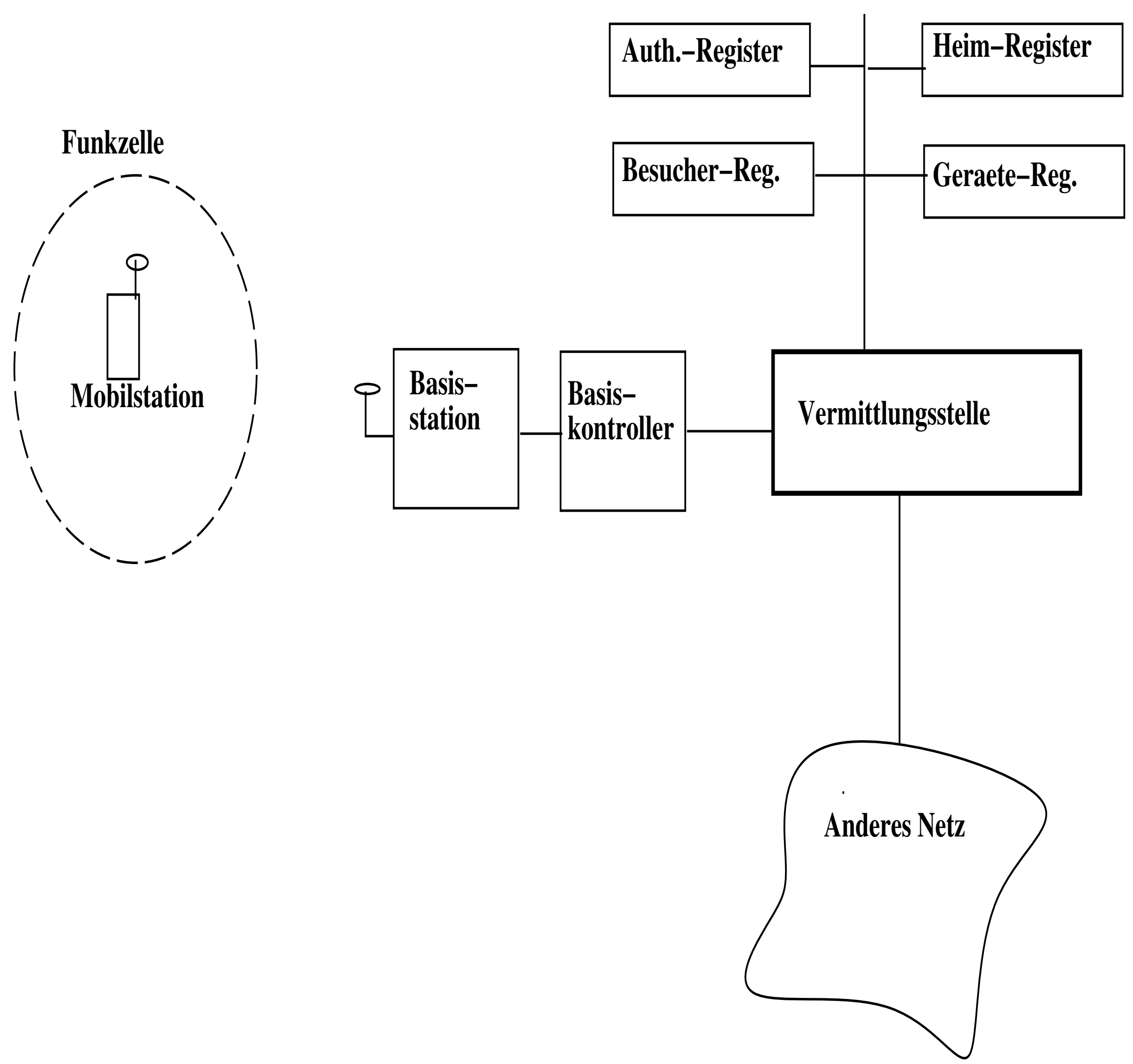


Abbildung 2.15: Archtitektur des GSM-Netzes

Das GSM-Netz hat eine Übertragungsrate von 300 bis 9600 bs und unterstützt Sprachübertragung, Paging und Datenübertragung.
Ein GSM-Netz besteht aus kleinen Funkzellen, die eine maximale Reichweite von 35 km haben. Die wesentlichen Komponenten des Netzes sind:

- *Die Mobilstation:*
  Ist das Endgerät mit dem der Zugang zum Netz ermöglicht wird, zu jedem Endgerät gehört eine Chip-Karte, die einen Teilnehmer-Identifikator sowie Sicherheitsfunktionen enthält.

- *Die Basisstation:*
  Ist für eine Funkzelle zuständig, sie empfangt und gibt Informationen weiter.

- *Der Basiskontroller:*
  Steuert die Basisstation.

- *Die Register:*
  Sind verteilte Datenbanken, die für die Speicherung von Benutzerdaten, Bewegungsdaten, Gerätedaten und Authentifizierungsdaten geeignet sind.

- *Die Vermittlungsstelle:*
  Ist eine Menge von Gateways. Sie vermittelt Verbindungswünsche an andere Vermittlungsstellen oder an einer Basisstation durch Festverbindungen. Sie unterstützt Verbindungen aus dem oder in das Festnetz und übernimmt die Verwaltungsaufgaben mit Hilfe der Register.

Die GSM-Zellen sind wabenförmig, um Überlappung von benachbarten Zellen zu vermeiden. Die Funkfrequenzen, die von einer Funkzelle belegt sind, können dadurch von anderen Funkzellen benutzt werden. GSM unterstützt das *Roaming*-Verfahren (Die Mobilstation kann die Funkzelle wechseln auch während einer Verbindung), was bedeutet, daß der Aufenthaltsort sowie auch die Funkfrequenz während **einer** Verbindung variabel sein können.
Das GSM-Standard unterstützt folgende Management-Aufgaben:

- Verwaltung von Aufenthaltsinformationen.

- Die Identifikation und die Überprüfung der Zugriffsberechtigung eines Benutzers

- Management von Funkkanälen (Kontroll- und Verkehrskanälen) u.a. die Überwachung und Bewertung der Kanalqualität.

- Roaming-Unterstützung.

Kapitel 3

Einsatzszenarien von Messagingsystemen im mobilen Umfeld

3.1 Einführung

In diesem Kapitel werden technische Einsatzszenarien von Messagingsystemen im mobilen Umfeld erläutert und in typischen Beispielen erklärt. In den folgenden Abschnitten wird besonders auf den Einfluß von Mobilität geachtet.
Einige Begriffe, wie der Zugriff auf Remote Mailboxen und die Eigenschaften von Mobile Messaging, werden zunächst erklärt, um die Szenarien beurteilen zu können.

3.1.1 Mobile Messaging

Mobile-Messaging läßt sich nach den verschiedenen Arten ([AB94], [AB93], [DBP94]) von Mobile-Computing einordnen:

- **Wireless-Computing**
  Endsysteme sind mit einer Wireless-Verbindung (z.B: Infrarot, Radiofrequenz) mit ihrer Arbeitsumgebung verbunden.
- **Nomadic-Computing**
  Benutzer ändern ihre Arbeitsumgebung ständig. Sie können z.B von einem fremden Netz aus ihre Nachrichten verarbeiten, oder Nachrichten schicken. Das erfordert ein Minimum an die genutzten Ressourcen und ein Maximum an Sicherheit für das Benutzer-Endsystem.
- **Decoupled Computing**
  Endsysteme können, mit dem Einsatz von Techniken wie Cache und Re-

synchronisation, auch eingesetzt werden, wenn sie vom Netz abgekoppelt sind.

3.1.2 Zugriffsprotokolle

Zugriffsprotokolle können anwendungsspezifisch oder filesystemspezifisch sein ([Hel95]), die zuletzt genannten werden aber aus folgenden Gründen nicht berücksichtigt:

1. Es gibt viele solche Zugriffsprotokolle, sie variieren aber nach Rechnertyp.
2. Sie sind eng verbunden mit dem Betriebssystem.
3. Daten sollen in einem speziellen Format gespeichert werden (auf dem Server)
4. Dateien werden komplett transferiert so das Bandbreite-Management fast unmöglich ist.

Für Anwendungsspezifische Zugriffsprotokolle sind drei Modelle zu unterscheiden:

- **Offline-Modell** (Beispiel: UUCP, POP-3):
  Auch Store-and-forward Model genannt. Auf Klient-Wunsch werden Nachrichten von einem zwischengeschalteten Server zum Klient übertragen.
  Der Klient baut periodisch eine Verbindung zum Server auf, ladet alle Nachrichten, die für ihn bestimmt sind, speichert sie und löscht sie dann vom Server, direkt danach wird die Verbindung abgebaut.
  Dieses Modell eignet sich besonders für PCs und MACs, die nicht immer eingeschaltet sind. Das ist aber ungeeignet, wenn ein Benutzer die Nachrichten von verschiedenen Rechnern bearbeiten will. Für Textnachrichten mit kleinen Größen kann dieses Modell Mobilnetze wesentlich entlasten.

- **Online-Modell** (Beispiel: NFS):
  Nachrichten werden auf dem Server gespeichert und dort vom Klient manipuliert. Die Daten können nur auf dem Server gespeichert und bearbeitet werden. Eine Verbindung zum Server ist während der gesamten Sitzung notwendig.
  Vorteile dieses Modells sind u.a.: Bessere Nutzung von schmalen Bandbreiten, Verlagerung der Funktionalität auf der Server-Seite, wenn der Klientrechner nicht über genug CPU und I/O-Kapazitäten verfügt.

- **Disconnected-Modell** (Beispiel: PCMAIL)
  Auch Cache-Modell genannt. Der Klient ladet bestimmte Nachrichten vom Server, macht eine Cache-Copy und baut die Verbindung ab. Nachrichten

	Offline	Online	Disconnected
Multiple-Clients	-	+	+
Sitzungsdauer (Minimal)	+	-	+
Nutzung Server-Ressourcen (Minimal)	+	-	-
Nutzung Client-Ressourcen (Minimal)	-	+	-
Multiple-Mailboxen (Remote)	-	+	+
Schneller Start	-	+	-
Mail-Bearbeitung wenn nicht Online	+	-	+

Tabelle 3.1: Ein Vergleich der verschiedenen Modelle

werden dann auf dem Client-Rechner, zu einem späteren Zeitpunkt wird die Verbindung wieder aufgebaut um mit dem Server die Veränderungen mitzuteilen.
Vorteile dieses Modells sind u.a.: Die Entlastung des Netzes und die Möglichkeit Nachrichten zu bearbeiten im Fall eines Server-Schutdownes

Ein Vergleich aus dem [RFC1733] verdeutlicht nochmals den Unterschied in der Tabelle 3.1:

3.2 Einsatzszenarien

3.2.1 Einfaches Emailsystem: SMTP

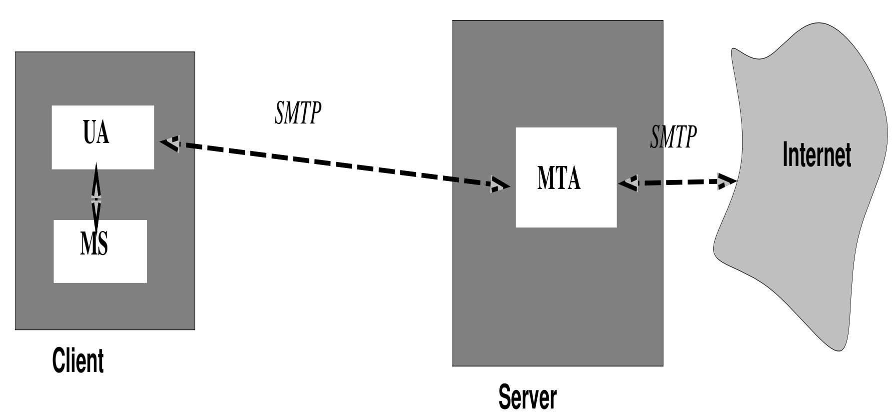


Abbildung 3.1: Einsatz von SMTP als Email-Protokoll

Eine Nachricht wird vom Benutzer erzeugt und dem UA weitergegeben. Der UA gibt die Nachricht an dem MTA weiter. Die Wahl des nächsten MTA oder Ziel-UA geschieht gemäß der angegebenen Adresse und einer Routing Tabelle. Der

Message Store ist implementierungsabhängig. Die Sicherheitsaspekte für SMTP sind in PEM (siehe Kapitel 2) diskutiert worden.
Der Einsatz von diesem Modell in Mobilumgebungen ist mit vielen Problemen verbunden:

1. UAs müssen ständig eingeschaltet sein, was unmöglich in einem mobilen Umfeld ist.
2. Temporär ausgeschaltete Rechner führen zu einer großen Netzbelastung wegen den wiederholten Zustellversuche.
3. Verlust von Nachrichten im langen Disconnected-Mode
4. Überflüssiger Nachrichten Transfer in verschiedenen Fällen:
   (a) Die Nachrichten sind zu groß
   (b) Rechner unfähig Graphiken oder Multimedia-Daten zu bearbeiten.
5. Bandbreitemanagement ist schwer, auch dann wenn SMTP-Erweiterungen die Nachrichtengrößen überprüfen.
6. Die Übertragungssicherheit ist nur durch externe Mechanismen möglich

3.2.2 Einfaches Emailsystem: X.400

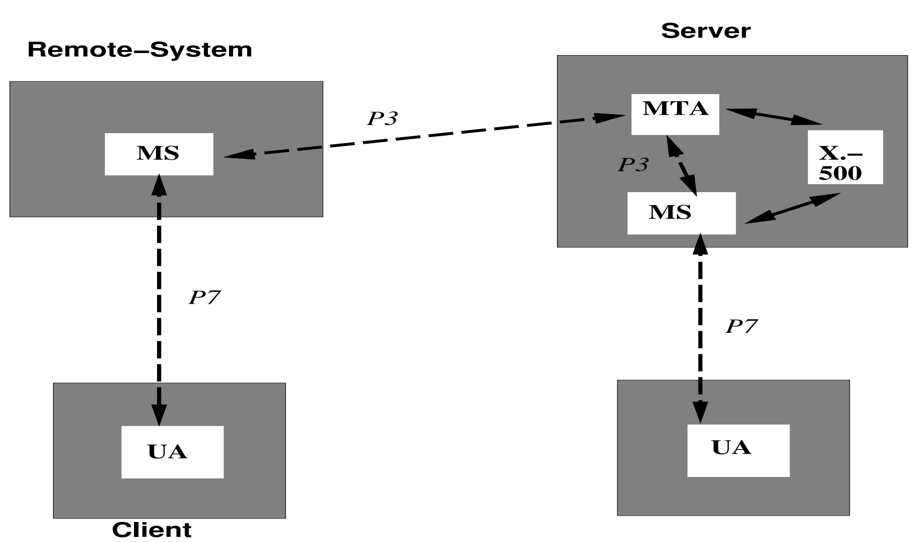


Abbildung 3.2: Einasatz von X.400 als Email-Protokoll

X.400 ist ein komplexes Protokoll. Der Nachrichtenverlauf ist wie bei SMTP. Zusätzlich ist ein Message Store (MS) zwischen UA und MTA geschaltet, das hat

den Vorteil, daß die Nachrichten nicht verloren gehen, wenn UAs nicht geschaltet sind (Vergleich voriges Szenario). Der MS gilt als Puffer zum UA. Er nimmt Nachrichten vom MTA, archiviert sie und auf Wunsch dem UA weiterleitet.
Eine sehr wichtige Komponente dieses Emailsystems ist das X.500 Verzeichnis, auf dem alle anderen Komponenten zugreifen können. Wegen der zwischengeschalteten MTA ist der Einsatz von diesem Szenario in einer mobilen Umgebung wesentlich besser als der erste, er ist vergleichbar mit der Kombination (SMTP, POP-3).
X.400 hat aber bekanntlich folgende Probleme:

1. Nachrichtenköpfe (z.B Adressen) sind zu kompliziert für den einfachen Emailnutzer, dessen Anzahl mit dem Einsatz der Mobilität zugestiegen ist, und auch für vielen Netzmanager.
2. Der Transport-System ist im Vergleich mit SMTP zu kompliziert und schwer zu implementieren, X.400 wird nur zum Teil implementiert (z.B: Verzicht auf P2)

3.2.3 Mail-Abhol-Protokoll: POP-3 mit SMTP

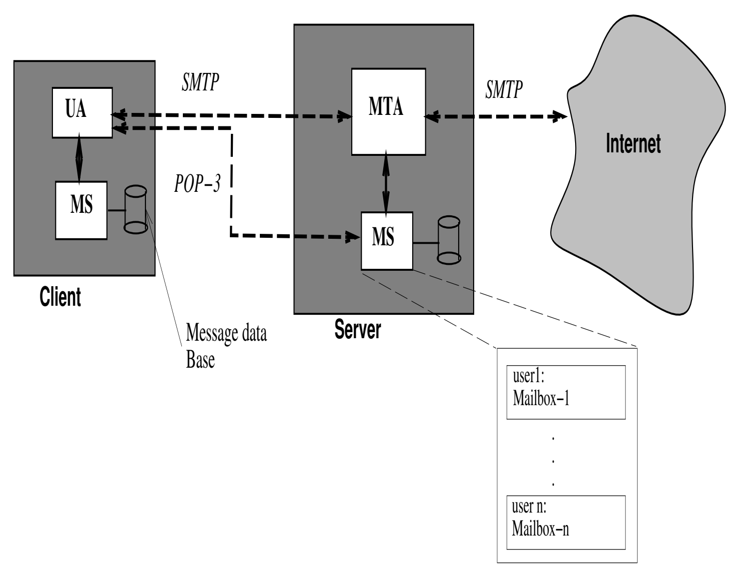


Abbildung 3.3: Einsatz von SMTP mit POP-3

1. *Senden*
   Für das Senden von Nachrichten wird SMTP verwendet. SMTP ist ein einfaches Protokoll, das auch im mobilen Umfeld eingesetzt werden kann. Es soll aber auf folgendes geachtet werden:

(a) Sicherheitsmechanismen sollen mehr beachtet, auch wenn es extern sein muß

(b) Der Zugriff auf MTAs in fremden Netzen (Nomadic Computing) soll ermöglicht werden

2. *Empfangen*
Nachrichten werden zu einem einzigen Rechner (POP-3 Server) gesendet und jeder Benutzer kann seine Nachrichten unter der Verwendung von POP-3 abholen und lesen. POP-3 hat folgende wichtige Eigenschaften:

(a) Es wird nicht eingesetzt, um Mails zu versenden.

(b) Unterstützung von nur Offline-Modell, der Klient kann aber so konfiguriert werden, daß die Nachrichten vom Server nicht gelöscht werden, wenn sie abgeholt worden sind. Diese Nachrichten werden aber bei jedem neuen Versuch, Nachrichten zu holen, mitgeschickt.

(c) Unterstützt den Zugriff auf neue Mail von jedem Ort im Netz.

(d) Einsatz von persistenten Nachrichten-Identifikatoren für das Disconnected-Modell: Nachrichten müssen in ihren gesamten Lebensdauer eindeutig identifizierbar sein, um die Synchronisation (Client-Server) zu erlauben.

(e) Das Datei-Format auf dem Server ist irrelevant für den Klient.

(f) Mit POP-3 können Nachrichten-Status nicht abgefragt oder verändert werden

(g) Multiple-Mailboxes werden nicht unterstützt (Vergleich IMAP-4).

(h) Archive-Mailboxen können nur lokal eingerichtet werden, das ist ein wesentlicher Nachteil von POP-3. Mobile-Messaging heißt u.a. die Nutzung von verschiedenen Klienten in verschiedenen Orten um Emails zu verarbeiten.

3.2.4 Mail-Abhol-Protokoll: IMAP-4 mit SMTP

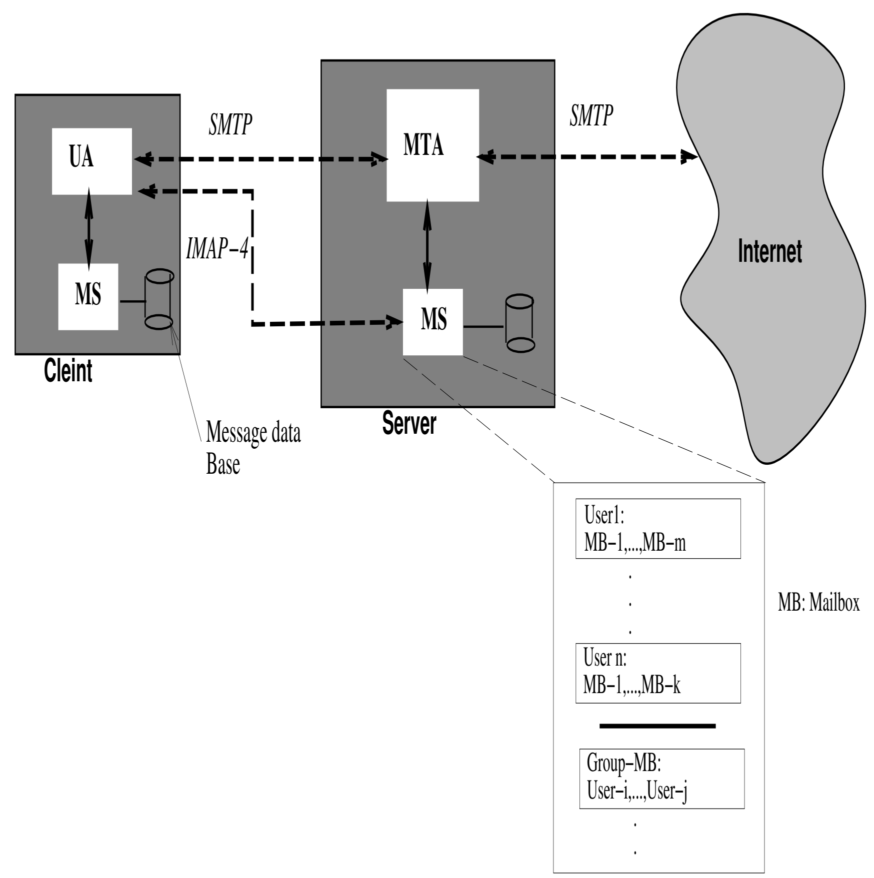


Abbildung 3.4: Einsatz von SMTP mit IMAP-4

1. *Senden*
   Es wird auch SMTP eingesetzt (siehe voriges Szenario).

2. *Emfangen*
   Nachrichten werden wie bei POP-3 zu einem einzigen Rechner (IMAP-4 Server). Eine Erweiterung um Multiple-Server wird zur Zeit der Erstellung dieser Diplomarbeit diskutiert. IMAP-4 hat mehr Vorteile im Vergleich mit anderen Protokollen um in einer mobilen Umgebung eingesetzt zu werden. Einige davon werden im folgenden aufgelistet:

   (a) *Bearbeitung von Remote-Mailboxen*

      i. Nachrichten remote von einem Mailbox in einem anderen abspeichern (z.B Einrichtung von Archive Mailboxen). Nachrichten sind immer verfügbar egal wo der Benutzer sich aufhält oder mit welchem Klient er zugreift.

ii. Unterstützung von Standard oder Benutzer-definierte Nachrichten-Flags in einer Mailbox: für den Benutzer ist das sehr wichtig, Nachrichten können z.B als wichtig, beantwortet oder eilig markiert. Für das Netz bedeutet es eine Entlastung durch die kürzere Sitzungen und bessere Nutzung der Bandbreite durch die Minimierung der übertragenen Datenmengen.

iii. Unterstützung von *Schared-Mailboxen* wie zum Beispiel ein Hilfe-Mailbox. Nachrichten-Flags können von verschiedenen Benutzern in verschiedenen Orten oder Zeiten manipuliert werden. Der Server sollte allen Benutzern über die Veränderungen zu jeder Zeit informieren!

(b) *Unterstützung von Multiple Mailboxen*
Wenn Filter für angekommene Nachrichten eingesetzt werden ist der Einsatz von verschiedenen Mailboxen sehr nützlich.

(c) *Online Leistungsoptimierung*

i. Nachrichten-Struktur abfragen, ohne die Nachricht zu laden.

ii. IMAP-4 unterstützt MIME-Nachrichten, mit IMAP-4 ist es möglich die Teile eine MIME-Nachricht abzufragen, einzelne Teile zu selektieren und nur gewünschte Teile zu laden.

iii. Unterstützung von einem Server-basierten Such- und Selections-Verfahren, Zweck ist die Minimierung der Daten-Übertragung, die Berücksichtigung der Client-Rechner-Kapazitäten und die Minimierung der Ressourcenbedarf auf der Klient-Seite.

(d) IMAP-4 unterstützt die drei Modelle: Online, Offline, Disconnected.

(e) Nomadic-Computing ist möglich.

(f) Sicherheit in IMAP-4 soll durch externe Mechanismen garantiert werden.

3.2.5 Kombiniertes Mail-System: IMAP-4, SMTP und ein Funksystem

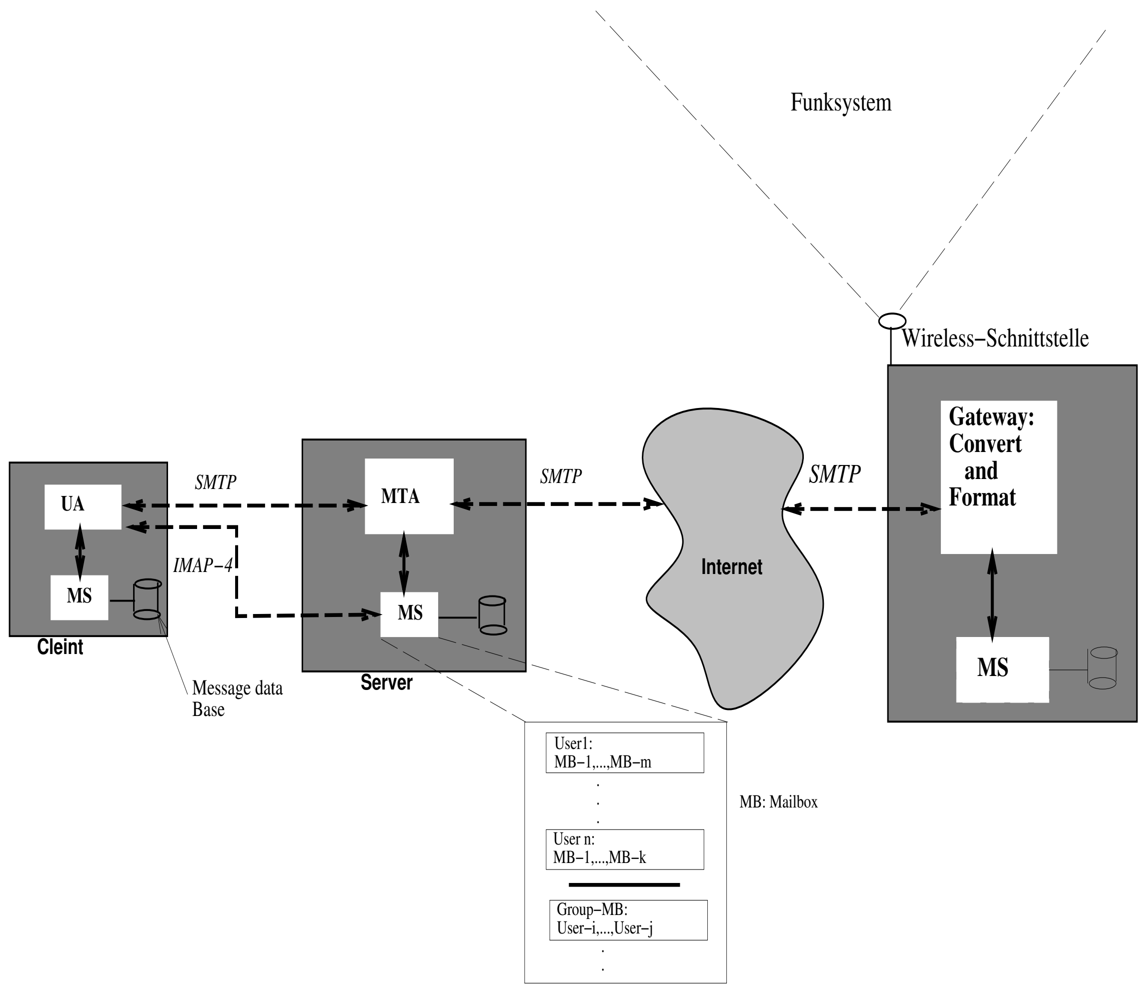


Abbildung 3.5: Einsatz von Gateways zu einem Funksystem

Wichtigste Komponente dieses Szenario ist auch die Gateway, die die Aufgabe der Konvertierung und Formatierung der Nachrichten übernimmt. Die Eigenschaften des Funksystems z.B Aufbau, Bandbreite, Übertragungsprotokoll, Sicherheitsmechanismen sind von großer Bedeutung für das Management.
Funksysteme sind dadurch charakterisiert, daß sie über schmale Bandbreite verfügen und eine große Fehlerrate haben. Der Aufbau und die Aufgaben der Gateway in diesem Szenario ist zu unterscheiden von Gateways, die verschiedene Email-Protokolle verbinden.

3.2.6 Paging: SNPP

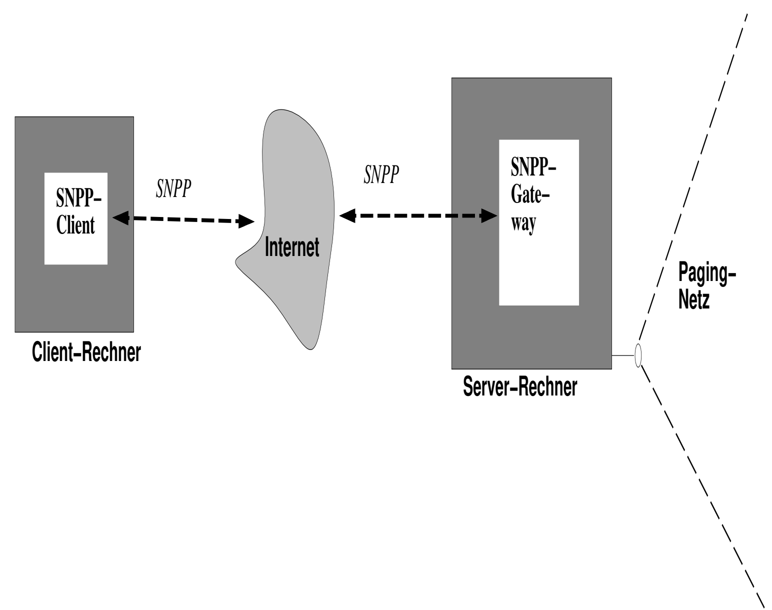


Abbildung 3.6: Einsatz von einem Paging-Dienst (SNPP)

Der Einsatz von SNPP ist besonders interessant für kurze Eilnachrichten. Die wichtigste Komponente ist das Gateway, das die Nachrichten konvertiert und formatiert und mittels einer Wirelessschnitstelle dem Paging-Dienst weiterleitet. Die Größe, Art, Aufbau der Nachricht und die Eigenschaften des Paging-Dienstes spielen dabei große Rolle für Management-Zwecke, Sicherheit wird wenig beachtet für diese Art des Messagings.

Kapitel 4

Messaging-Architektur: Komponenten und ihre Kommunikationsvorgänge

In **Kapitel 3** wurden die Messaging-Protokolle anhand von Szenarien diskutiert und miteinander verglichen. In diesem Kapitel wird eine Messaging-Architektur, die sich für das mobile Umfeld mit Hinblick auf die Ergebnisse vom vorigen Kapitel angegeben und auf einem System-Modell abgebildet.
Ziel dieses Kapitels ist, die zu verwaltenden Komponenten zu spezifizieren um einen Management-Ansatz darzustellen. Die Unterscheidung der Komponenten erfolgt nach der Definition ihrer Funktionalität, was einen größeren Teil dieses Kapitels auszeichnet. Weiter werden die Kommunikationsvorgänge der einzelnen Komponenten aufgezeigt.

4.1 System-Modell

Der Design eines System-Modells wurde in verschiedenen Projekten erforscht (siehe [IB95], [AB94], [LMM95], [IDM91]). Die definierten Modelle basieren auf dem gleichen Prinzip und unterscheiden sich nur gering wie z.B. die Suche nach dem Aufenthaltsort eines Mobilrechners. Hier wird ein vereinfachtes Modell basierend auf [IDM91] und dem GSM-Modell ([JLLM94]) dargestellt.

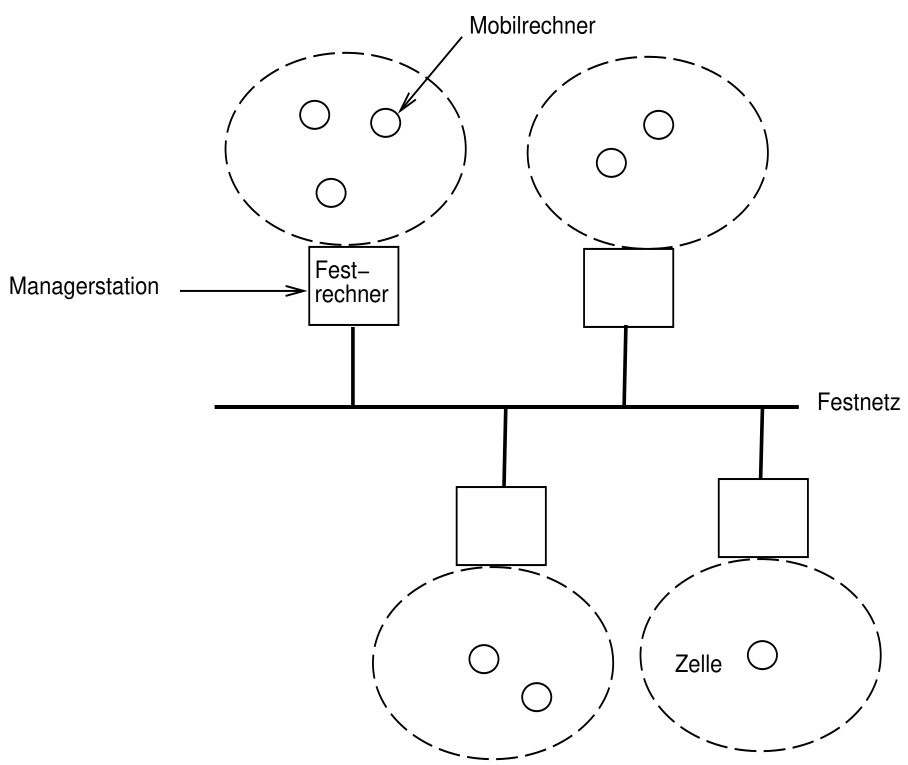


Abbildung 4.1: Ein vereinfachtes System-Modell

Ein Netz mit mobilen Endgeräten besteht im wesentlichen aus zwei Komponenten-Gruppen, eine Gruppe von Mobilrechnern [1], die in geographischen oder logischen Untergruppen genannt *Zellen* unterteilt ist. Jeder Mobilrechner identifiziert sich in einer einzigen Zelle.
Die zweite Gruppe bildet ein Festnetz, das aus Managerstationen (**MS**) besteht. Jeder Managerstation ist mit einer Zelle verbunden, sie dient der Kommunikation mit den, in der Zelle sich aufhaltenden Rechnern. Mobilrechner (**mr**), die sich in einer Zelle identifizieren werden als *lokal* zu der Managerstation betrachtet. Weiter wird in diesem Modell die Managerstation als Wireless-Schnittstelle betrachtet und es wird vorausgesetzt, daß ein Mobilrechner, der sich in einer Zelle aufhält nur mit der Managerstation dieser Zelle kommuniziert.

4.1.1 Nachrichtenverlauf

Aufbauend auf dieses Systemmodell und auf den GSM-Standard (siehe **Kapitel 2**) wird nun der Nachrichtenverlauf skizziert werden. Für Details und Erklärungen wird auch auf [JLLM94] verwiesen:

[1] Ein Rechner, der seinen Aufenthaltsort ohne Abbruch der Netzverbindung ändert, ist ein Mobilrechner ([IDM91])

Sei mr1 ein Mobilrechner, der einen anderen Mobilrechner mr2 eine Nachricht [2] senden will. mr1 sendet durch eine Wireless-Verbindung die Nachricht zur lokalen Managerstation MS1, dann können verschiedene Szenarien auftreten:

1. mr2 hält sich lokal in seiner Zelle auf, die Nachricht wird dann zu der Managerstation dieser Zelle zugeschickt, die sie dann zu mr2 weiterleitet.

2. mr2 hält sich fremd auf, dann soll er zuerst gesucht werden, was auch mit einem Routing-Problem verbunden ist (siehe [IDM91]), wenn die Routing-Tabellen statisch sind. Der Suchvorgang wird im Folgenden mittels das GSM-Systems (siehe **Kapitel 2**) erklärt. Zuerst mit dem einem Pseudo-Code ([JLLM94], [IB95]):

```
                  { call to mr2 is detected at local
                  MS1;
                  if called party is in the same cell
                  then return;
                  MS1 queries called party's Home
                  Location Register (HLR);
SEARCH()          called party HLR queries called
                  party's current Visitor Location
                  Register (VLR), V;
                  V returns called Party's location to
                  HLR;
                  HLR returns location to calling MS;
                  }
```

   Nach diesem Beispiel wird zuerst nachgefragt ob *mr2* in der selben Zelle wie *mr1*, falls nicht wird der Home Location Register von *MS2* abgefragt, dieser enthält einen Zeiger auf dem Visitor Location Register (VLR), in dem sich momentan *mr2* aufhält. Der VLR fragt *MS2*, ob *mr2* momentan Nachrichten empfangen kann, oder ob er vom Netz abgekoppelt ist. *MS2* übermittelt VLR dann eine gültige Routing-Adresse, die dann an *MS1* weitergeleitet wird.
   Erwähnungswert ist in diesem Zusammenhang die in [JLLM94] vorgeschlagene Cache-Strategie: Jeder MS verfügt über einen lokalen Speicher, genannt *Cache*, um Aufenthaltsinformationen über den mobilen Rechnern zu speichern. In dem Beispiel durchsucht *MS1* zuerst ihren Cache, ob Aufenthaltsinformationen über *mr2* vorhanden sind, falls nicht, findet der oben erklärte Vorgang statt. Der Pseudocode sieht dann folgendermaßen aus:

[2] nicht zu vertauschen mit Mail-Nachrichten, dies wird im nächsten Abschnitt behandelt

`CacheSEARCH()`	`{ call to mr2 is detected at local MS1;` `if called party is in the same cell then return;` **if there is no cache entry for called mr2 then invoke SEARCH() and return;** `MS1 queries VLR, V specified in the cache entry;` `if called mr2 is at V, then V returns party's location to calling MS` `else { V returns miss to calling MS; calling MS invokes SEARCH();` `} }`

4.2 Messaging-Architektur

Auf dem im vorigen Abschnitt vorgestellten Systemmodell wird eine Messaging-Architektur, die sich im mobilen Umfeld eignet (Siehe **Kapitel 3**), abgebildet.

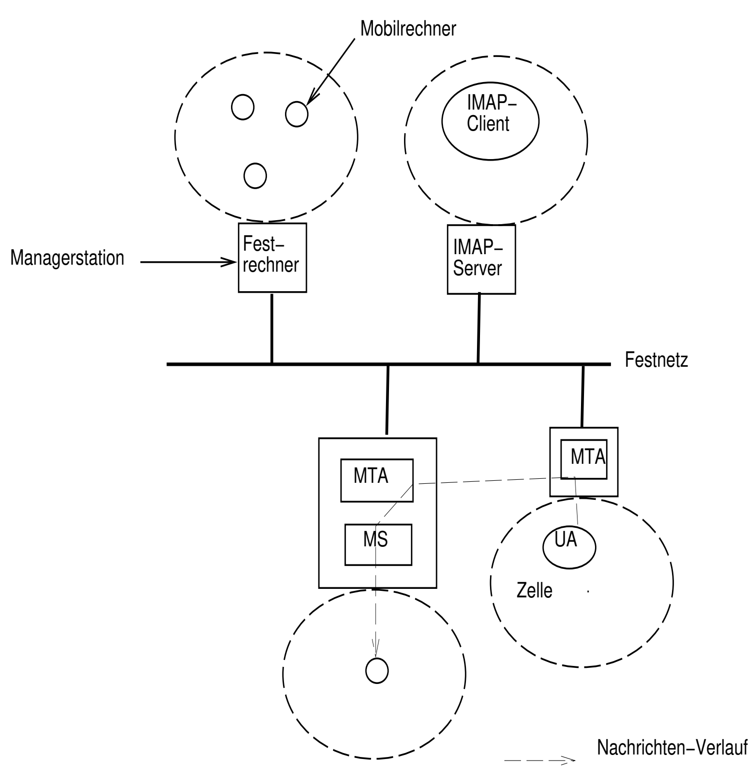


Abbildung 4.2: Messaging-Architektur

Es wird im weiteren Verlauf der Diplomarbeit vorausgesetzt, daß eine IMAP-4 oder ähnliche Architektur angewendet wird, um die Mobilitätsanforderungen gerecht zu werden.
Mobilrechner sind mit IMAP-4-Klienten ausgerüstet. Diese kommunizieren mit einem Server auf einem Festrechner (zur Vereinfachung wird hier die Manager-Station als Server und Wireless-Schnittstelle betrachtet).
Empfangene Nachrichten werden auf dem Server gespeichert, um zu einem späteren Zeitpunkt abgeholt zu werden oder direkt durch einen SMTP-MTA dem Mobilrechner zugeschickt werden.

4.2.1 Messaging-Komponenten

Beschreibung und Funktionalität

In 4.3 sind die Komponenten eines Messaging-Systems skizziert.

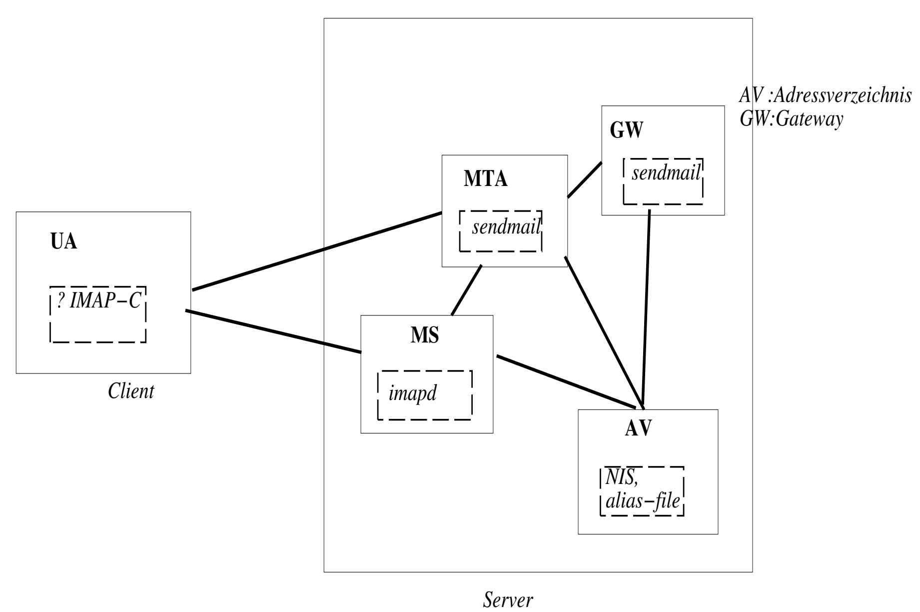


Abbildung 4.3: Komponenten eines Messaging-Systems

- **Der User Agent (UA)**:
  Der UA ist eine Menge von Werkzeugen:
  1. Ein Editor, um Nachrichten zu editieren.
  2. Ein Lese-Werkzeug, um Nachrichten repräsentieren und lesen zu können.
  3. Ein Sende-Werkzeug, der die Kommunikation mit dem Message Transfer Agent unterstützt, um Nachrichten senden zu können.

4. Ein Hole-Werkzeug, initiiert die Kommunikation mit dem Message Store und dient der Manipulation von Nachrichten. Dieses Werkzeug soll über eine gewisse Intelligenz verfügen um z.B Cache-Verfahren einzusetzen oder die Benutzer-Tätigkeiten und Eigenschaften aufzunehmen und zu analysieren.

Die Aktivitäten des UAs sind zu unterscheiden von denen des Servers. Verschiedene Performance-Tools wurden entwickelt, um den Klient zu unterstützen (siehe z.B [LMM95]). Obwohl Klienten eine große Bedeutung in einem mobilen Umfeld haben, werden im weiteren nur die anderen Komponenten betrachtet, weil sie einen wichtigeren Management-Bedarf haben.

- **Message Transfer Agent (MTA)**:
  Speichert die Nachrichten vorübergehend. Der MTA übernimmt die Routingaufgabe, empfängt und sendet Nachrichten weiter, dupliziert Nachrichten und sendet sie weiter, falls es sich um eine Liste handelt und konvertiert die Nachrichten, falls erforderlich. MTA's können Nachrichten ablehnen z.B, wenn sie eine bestimmte Größe überschreiten.
  Es besteht die Notwendigkeit die Routing-Pfade dynamisch zu wählen (siehe **Abschnitt 4.1***), dadurch lassen sich auch die Nachrichtendringlichkeit und Verfügbarkeit von Verbindungen berücksichtigen.* Die Hauptaufgabe des MTA's ist also das Routing von Nachrichten, das ist in heutigen Messagingsystemen von verschiedenen Eigenschaften und Möglichkeiten geprägt:

  1. Dynamische Routing: ist notwendig in einer mobilen Umgebung.
  2. Listen-Routing: Empfänger der Nachrichten können vom Sender hintereinander in einer Liste eingegeben werden, ein Empfänger wird erst die Nachricht bekommen, wenn es vom vorigen Empfänger genehmigt (beziehungsweise weitergeleitet) wird.
  3. Performance: Schnelle und günstige Zustellwege herausfinden.

  MTA's übernehmen sowohl die Rolle des Empfängers als auch des Senders. Dabei können Probleme wie Staus in Nachrichtenschlangen und sowie Loop-Schleifen der Nachrichten auftreten.

- **Message Store (MS)**:
  Der Message Store spielt die Rolle eines permanenten Nachrichtenspeichers. Gewünschte Nachrichten werden im MS aktiv gesucht. Im MS werden Nachrichten in Mailboxen gespeichert, die Mailboxen können *privat, für eine Gruppe, Archiv oder fremd sein*, das hängt aber vom eingesetzten Messagingsystem ab. Mailboxen werden durch Identifikatoren erkannt und ein Benutzer kann eine oder mehrere Mailboxen haben. Benutzer- Nachrichten und Mailboxen können permanent im Server gespeichert werden, einige Messagingsysteme bieten aber die Option, um die Nachrichten lokal

oder auf einem anderen Speichermedium zu verschieben. Benutzer, die auf dem Message Store zugreifen können werden vom Server-Manager eingerichtet. Nachrichten können mit Hilfe einer Datenbank oder einem Filesystem gespeichert werden. Sie können im Message Store verschlüsselt gespeichert werden. Nachrichten sind je nach Police für die Manager lesbar oder unlesbar. Relevante Informationen für den MS können u.a. Größe, Alter, Priorität, Vertraulichkeitsgrad von Nachrichten sein. Dieses ist in den meisten Fällen für Server-Manager zugreifbar um gewisse Management-Entscheidungen treffen zu können (siehe nächstes Kapitel).
Zusammenfassend läßt sich der Message Store wie folgendes definieren:

1. Eine bestimmte Anzahl von gleichartigen oder unterschiedlichen Mailboxen mit einer je nach vorhandenen Speicherkapazitäten frei definierbaren Speichergröße.
2. Ein Speichermedium der sich auf dem Server, lokal beim Remote-Benutzer oder auch ein anderes, im Netz (Festnetz) vorhandenes Speichermedium, befindet.
3. Eine unbestimmte Anzahl von Nachrichten mit bestimmten Attributen und Eigenschaften.
4. Eine vom Server-Manager eingerichtete Menge von Benutzern, die auch temporär sein können (z.B Fremd-Benutzer, Gast-Benutzer).
5. Kommunikationsschnittstellen zum UA, MTA, AV und auch Anwendungen, die messagingfähig sind.
6. Plattformspezifische Eingenschaften, die die Unterstützung der unterschiedlichen Plattformen (UNIX, Windows) dienen.

Der MS ist eine wichtige Komponente in einer mobilen Umgebung. Nachrichten werden eingespeichert und verwaltet bis sie abgeholt oder gelöscht werden. Konventionelle Implementierungen von Mail-Systemen stutzen sich auf Dateisysteme, um dem MS zu realisieren.

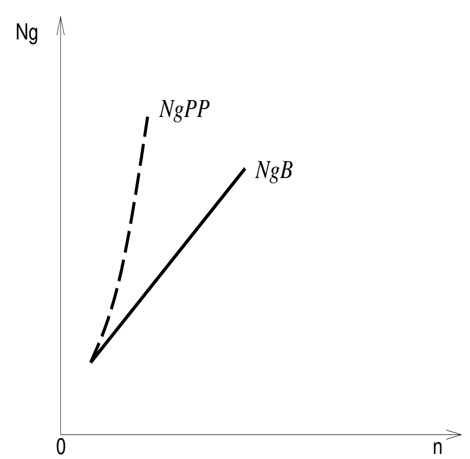
- **Die Nachricht**
  Die obengenannten Komponenten müssen entweder Nachrichten austauschen oder enthalten und verarbeiten (z.B : Speichern oder store-and-forward). Nachrichten können:
  - Benutzer-Nachrichten
  - Nachrichtenverlauf Status-Berichte oder
  - Anwendungsspezifische (zwischen Anwendungen ausgetauschte) Nachrichten sein.

Nachrichten können sich in Art, Aufbau, Größe,... unterscheiden. Sie können auch im gleichen Protokoll unterschiedlich sein (Vergleich X.400), SMTP-RFC822 unterscheidet nicht zwischen Nachrichten. Erweiterungsvorschläge können aber z.B Report-Nachrichten unterscheiden. Nachrichten haben eindeutige Identifikatoren aber nicht in allen Messaging-Protokolle.
Eine Nachricht hat in der Regel zwei Teile: der Nachrichtenkopf und der Nachrichtenrumpf und können unterschiedliche Formate, die proprietär oder Standard sein können, haben. Nachrichtenformate sind mehr unterschiedlich als einheitlich. Benutzer sind aber in der Regel auf Formate, mit denen sie vertraut sind, angewiesen.
Die Nachrichtenformat-Elemante sind:

1. Text und Zahlen
2. Konventionen für die Typen der Attachements (Binär, Text, Acrobat,...)
3. Konventionen für die Codierung der Attachements (MIME Gif, ANCII Text, BinHex, ...)
4. Konventionen für die Benennung der Dateien (8, 20, ... Buchstaben)
5. Konventionen für den Nachrichtenkopf: TO, FROM, DATE, SUBJECT, CC, BCC,...
6. Konventionen für Nachrichten-Attribute: Priorität, Datum und Zeit, Identifikator, Vertraulichkeitsgrad, Verschlüsselung, Wichtigkeitsgrad, Bestätigungsinformation (angekommen, gelesen, ...), Verbindlichkeitsinformation (z.B nach 2 Stunden nicht mehr verbindlich)
7. Konventionen für Routing-Informationen (z.B Adressen)
8. Konventionen für Aktionen: Trigger, replay, replay-to-replay, forward, followup, ...

- **Gateways**
Gateways werden in heterogenen Umgebungen eingesetzt. Sie konvertieren Nachrichten von einem Nachrichtenformat eines Messagingsystems zu einem anderen. Gateways sind meistens mit MTA's auf dem gleichen Rechner installiert.
Gateways werden in existierenden Management-Ansätze als spezielle MTAs angesehen. Gateways können auch eingesetzt werden, um Nachrichten eines Messaging-Dienstes (z.B. Email) in einem anderen (z.B. Paging) zu konvertieren.
Man unterscheidet zwischen *Point-to-Point* Gateways zwischen verschiedenen Messagingsystemen, wobei ein Messagingsystem eine solche Gateways pro unterschiedlichen Messagingsystem benötigt und *Backbone-Lösungen*

(siehe Abbildung unten), die die Anzahl der Gateways wesentlich reduzieren, indem nur eine Gateway zu einer Backbone benötigt wird.
Der Unterschied zwischen diesen Ansätzen wird im folgenden Beispiel demonstriert:
Man betrachte eine Firma mit n Standorten, die unterschiedliche Messagingsysteme (SMTP, X.400, MSMail, ...) nutzt. Sei $NgPP$ die Anzahl der benötigten Gateways bei einem *Point-to-Point*-Ansatz und NgB die Anzahl der Gateways bei einem *Backbone*-Ansatz. Um den Unterschied zu zeigen, werden im folgenden diese beide Zahlen berechnet und graphisch dargestellt werden:
$NgPP = n * n$
$NgB = n + 1$ *(Backbone)*



Eine Gateway oder Gateway-Backbone hat denn die folgenden Funktionen:

1. Nachrichten Transfer
2. Konvertierung der Nachrichten-Formate
3. Konvertierung der Adressen

Es ist zu bemerken, daß eine Backbone zwar verschiedene Formate vereinheitlicht, aber diese wieder in einem neuen Format (Adressen und Nachrichten) konvertiert, das soll aber ein wohldefinierter Standard wie SMTP oder X.400 sein.

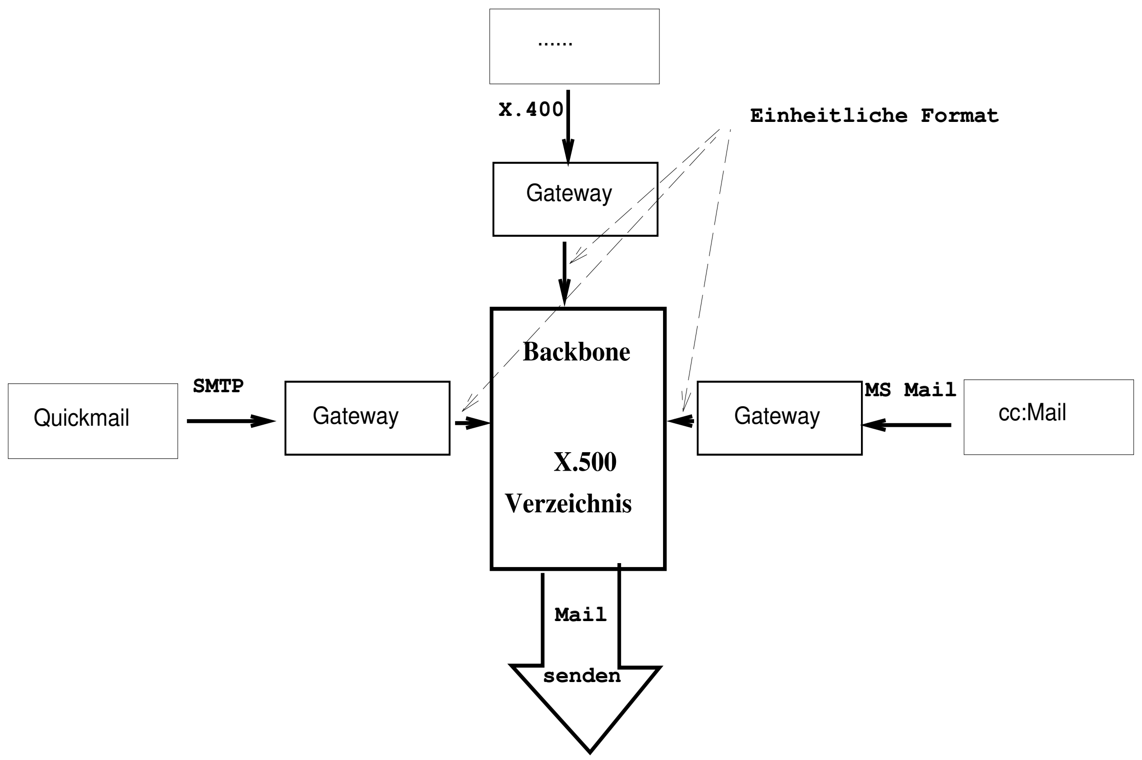


Abbildung 4.4: Gateways zu einer Backbone mit dem X.500 Verzeichnis

- **Adreßverzeichnis (AV)**
  Das Adreßverzeichnis ist eine Schlüsselkomponente in einem Messagingsystem, egal ob es proprietär oder Standard ist und besonders auch in heterogenen Umgebungen. In jedem Messagingsystem, ist ein Adreßverzeichnis mit den Namen und Adressen aller Benutzern enthalten. Ein Benutzer, der eine Nachricht senden will, wählt den Namen und Adresse des Empfängers aus dem Adreßverzeichnis.
  In einem Messagingsystem wird im allgemeinen jedem Domain oder einer Gruppe von Benutzern ein Adreßverzeichnis zugeordnet. Eine Änderung in einem Adreßverzeichnis soll in den anderen eingetragen werden. Heutige Messagingsysteme bieten dafür automatische Tools, die die Synchronisation vornehmen. Die Schwierigkeit liegt aber darin, wenn mehrere, unterschiedliche Messagingsysteme zum Einsatz kommen. Netzmanager sollen meistens manuelle Änderungen führen, um die Adreßverzeichnisse zu synchronisieren. Die Lösung ist ein globales Adreßverzeichnis (z.B. X.500) der u.a. folgende Information enthalten soll:

  1. Benutzer-Namen
  2. Benutzer-Adressen

3. Spezifische Information über Benutzer, kann auch freidefinierbar sein
4. Nachrichtenformat-Information, insbesondere für proprietäre Messaginsysteme
5. Adressenformat-Information
6. Routing-Information
7. Sonstige Information wie z.B. Attachement-Konventionen.

Kommunikationsvorgänge

- **UA-MS**
  Die Verbindung UA-MS ist bidirektional und hat eine Schlüsselrolle in einer mobilen Umgebung. Die ausgetauschten Daten können sich in Größe, Typ, Dringlichkeit, Vertraulichkeit und Zusammensetzung unterscheiden. Sie können u.a. kurze Mitteilungen, Multimedia-Daten, Server-Antworten, Klient-Abfragen sein. Dabei spielt das Verbindungsmedium und das Netzart eine große Bedeutung, diese beeinflussen wesentlich die Verbindung, Art und technische Eigenschaften dieser Verbindung sind:

  1. Local Area Networks (LAN's)
  2. Mobile Local Area Networks, wie z.B IEEE 802.11 mit einer Bandbreite von 2Mbps.
  3. Wide Area Networks (WAN's)
  4. Mobile Wide Area Networks wie Cellular Digital Packet Data (CDPD) mit einer Bandbreite von 11.8 Kbps (Downlink) und 13.3 Kbps (Uplink), und Cellular VLR/HLR (z.B GSM) mit einer typischen Bandbreite von 9.6 Kbps.

- **UA-MTA**
  In der festgelegten Architektur ist diese Verbindung in der Richtung MTA zu UA fast überflüssig, da der MTA die Nachrichten im Message Store ablegt. In der anderen Richtung (UA zu MTA) kommuniziert der UA mit dem MTA, um Nachrichten zu versenden. Die technischen Eigenschaften dieser Verbindung sind die gleichen wie diesen UA-MS.

- **MTA-MS**
  Der MTA kommuniziert mit dem MS um Nachrichten zu übermitteln. Nachrichten können für ein Benutzer, für eine Liste von Benutzern oder für eine Gruppe sein. Die Eigenschaften dieser Verbindung sind unterschiedlich von der UA-MS, weil sie im Festnetz stattfinden.

- **MTA-AV**
  Der MTA kommuniziert mit dem AV um Routinginformation zu holen, das aber nicht immer der Fall, die Routinginformation ist in vielen Messagingsystemen der Adresse enthalten. Der MTA könnte auch Aufenthaltsinformation eines Benutzers vom AV extrahieren, was über ein geeignetes AV bedarf. Die technischen Eigenschaften dieser Verbindung sind die eines Festnetzes.

- **UA-AV**
  Der UA kann das AV nutzen um Informationen über einzelnen Benutzer zu holen, ein Beispiel dafür ist die Ermittlung der Adresse eines Benutzer mit Namenseingabe. Die technischen Eigenschaften dieser Verbindung sind die gleichen wie UA-MS.

4.2.2 Objektmodell des Message Store

In diesem Abschnitt werden die Komponenten des Message Stores genauer behandelt, da sie für den weiteren Verlauf der Diplomarbeit wichtig sind (siehe Abschnitt 6.2).

- Eine Menge von Mailboxen, die Einzelner, oder Gruppen von Benutzern gehören können. Mailboxen können sich in einem, oder mehreren Orten befinden. Benutzer können je nach Messagingprotokoll einen, oder mehrere Mailboxen haben. Mit IMAP-4 z.B ist es möglich Mailboxen lokal oder entfernt zu kreieren, löschen oder umbennen, zu einer Groupmailbox können sich Benutzer anschließen (subscribe), oder auch entfernen (unsubscribe).

- Mailboxen bestehen aus Nachrichten, die dann die gleichen Eigenschaften einer Mailbox haben, zusätzlich ihre eigenen, die sie kennzeichnen und wodurch sie identifizierbar sind. Nachrichten haben zwei grundliegende Eigenschaften, die man als dynamisch und statisch einordnen kann. Die dynamische Eigenschaft ist, daß die Nachrichten gesendet und empfangen werden können. Nachrichten haben ein Ursprung mit Ziel-Ort und wandern durch duzende von Stationen bevor sie am Ziel gelangen. Die statische ist das Ziel selbst. Dort werden sie an einen Ort, in eine Mailbox abgehalten.

- Ein kommunikationsfähige Prozeß : Kommunikationsvorgang besteht aus verschiedenen Teilnehmer, ausgetauschten Informationen und Kommunikationsmedium. Diese drei Gruppen können sich in ihren Eigenschaften und Funktionalität unterscheiden.

- Benutzer oder Grouppen von Benutzern, denen Mailboxen gehören.

Zusammenfassend können sich 4 Objektgruppen unterscheiden (siehe Abbildung):

- MS-Domaine (imapd, pop3d,...)
- Mailboxen
- Nachrichten
- Verbindungen
- Benutzer

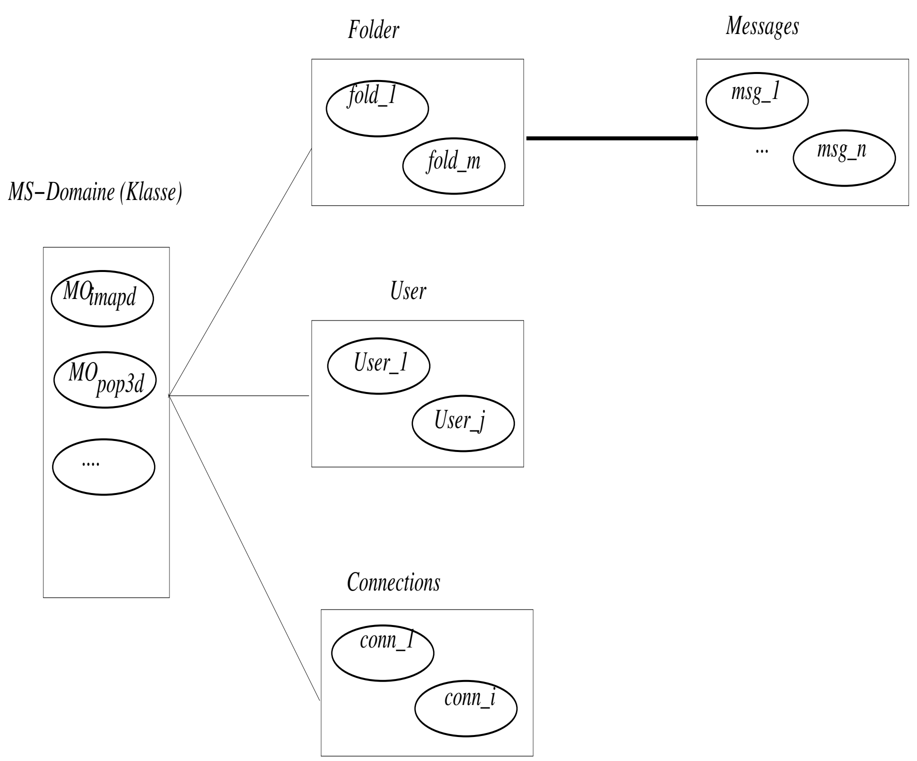


Abbildung 4.5: Objektmodell des Message Stores

Kapitel 5

Managementanforderungen

In Kapitel 1 und 3 wurden Szenarien zur Nutzung von Messagingsystemen aufgezeigt. Kapitel 4 beschäftigte sich mit den Komponenten einer vorgeschlagenen Messaging-Architektur. In diesem Kapitel sollen, die daraus entstehenden Managementanforderungen entlang der Management-Funktionsbereichen erläutert werden. Aus den Managementanforderungen wird die Managementinformation zum Teil abgeleitet und in Kapitel 6 vollständig dargestellt. Dieses Teil der Managementinformation bezieht sich auf den Message Store.

5.1 Rahmenbedingungen

Die Rahmenbedingungen, die aus dem Einsatz von mobilen Endgeräten resultieren, werden in diesem Abschnitt zusammengefaßt, um einen gesamten Überblick zu gewinnen. Mobilsysteme sind dadurch gekennzeichnet, daß die physische Struktur des Netzes variabel[1] ist, so daß die logische Struktur nicht auf der Menge der physischen Komponenten und Verbindungen basieren kann, Veränderungen treten nicht nur im Fall von Knoten- oder Verbindungsausfällen ein. Die Kommunikationsmedien sind sehr unterschiedlich, was auf die Übertragungszeit und die Funktionalität der Netzkomponenten einen großen Einfluß hat, und leicht abhörbar (hohe Sicherheitsanforderungen).
Die mobilen Endgeräte sind heterogen und durch begrenzte Speicher, CPU und Energiekapazitäten charakterisiert. Das hat verschiedene Folgen, wie die zeitweise Abkopplung, die Schutdowngefahr, die Unfähigkeit speicherintensive Algorithmen auszuführen, wie beispielsweise Such- und Sortierfunktionen, das erfordert besonders geringe Kommunikation und die Verlagerung der Funktionalität auf das Festnetz (Server).
Mobilrechner sind ebenfalls sogenannte bewegliche Knoten, die nur zeitweise als Endknoten innerhalb der Netztopologie sichtbar sind und in dieser Zeit auch bisweilen als Dienstnutzer fungieren. Die Ressourcen, die ihnen zur Verfügung

[1] Vergleichbar mit einer Variable, deren Inhalt sich mit jeder Bewegung ändert

stehen, ändern sich häufig. Durch ihre Beweglichkeit können sie auch ihre Zellen oder Domäne (siehe **Kapitel 4**) wechseln, nachdem sie einen Service verlangt haben und bevor sie eine Antwort bekommen.

5.2 Konfigurationsmanagement

5.2.1 Einleitung

Die eindeutige Identifizierung von Benutzern, Mailboxen und Nachrichten ist notwendig, um die Attribute dieser Objekte abzufragen, zu setzen oder zu ändern. Das Konfigurationsmanagement soll u.a. die Funktionstüchtigkeit der zu managenden Komponenten garantieren. Die Messaging-Komponenten, insbesondere MTAs und MSs, sollen durchgehend funktionsfähig sein. Die Konfiguration der Komponenten, insbesondere Gateways, ist ebenfalls von großer Bedeutung.

5.2.2 Anforderungen

1. **Konfiguration der Komponenten**
   Die Konfiguration von Gateways stellt zwar eine zentrale Rolle in einem Messagingsystem wird aber an dieser Stelle nicht betrachtet (siehe Abschnitt 4.2). Für den Message Store wird folgende Information benötigt:

   - Der Name des Services eines MSs, einige Beispiele sind: IMAP-4, POP-3, ...
   - Die Version des Services. Neuere Versionen können Erweiterungen enthalten.
   - Die Anzahl der Login-Versuche vor einem Time-Out, die entsprechend erhöht oder erniedrigt werden kann, je nach Netzverkehr.
   - Die Konfiguration von einzelnen Parametern wie Anzahl der Nachrichten, die bei einem Kommando z.B. **Fetch bei IMAP-4** (siehe Abschnitte 2.1.2 und 4.2.1) durchsucht werden müssen. Im nächsten Kapitel werden diese Parameter genauer im Zusammenhang mit der zu managenden Komponente spezifiziert.
   -

2. **Aufenthaltsverwaltung und Roaming-Unterstützung**
   Mobilrechner ändern ihren Aufenthalt ständig auch während einer Verbindung (siehe Abschnitt 4.1). *Dieses zu verwalten und die Roaming-Unterstützung* zu ermöglichen ist eine der Managementanforderungen. Diese Aufgaben werden mit den vorhandenen Techniken (z.B. Mobile-IP, siehe Abschnitt 2.5) auf Transportebene behandelt. Sie können aber anwendungsspezifisch

unterstützt werden. Die folgende Information wird u.a. benötigt (siehe auch Positionierung der Mailbox):

- Der momentane Aufenthalsort eines Benutzer, der eine Verbindung zum MS aufgebaut hat.
- Die Home-Adresse eines Benutzers

Die Information über die Bewegungen der Benutzer kann geloggt werden, um daraus Statistiken zu erstellen. Das kann eingesetzt werden um den Aufenthaltsort eines Benutzers zu prognostizieren, damit z.B. eine Eilnachricht automatisch verschickt werden kann (siehe Abschnitt 4.2.1).

3. **Überschreitung der Managementdomänen**
   Nachrichten überschreiten in vielen Fällen die Managementdomänen (siehe Abschnitt 4.1). Es soll ermöglicht werden, die Nachrichten auch dann verfolgen zu können. Hierbei spielt auch die Managementarchitektur eine wesentliche Rolle. Eine Manager-to-Manager-Kommunikationsmöglichkeit ist notwendig. Dafür wird dann u.a. folgende Information benötigt:

   - Die Adressen der Agenten der MTAs, von denen die Nachrichten zum Message Store weitergeleitet werden. Diese Information kann für die Verfolgung der Nachrichtenverlauf benutz werden: Rückwärts, um mögliche Fehler (siehe auch Fehlermanagement) zu lokalisieren, und auch aufwärts, um z.B. den nächsten Ziel-MTA zu ändern, wenn er untüchtig ist.

4. **Positionierung der Mailbox und Replikation der Nachrichten**
   Eine Nachricht kann z.B umgeleitet werden, nach dem Prinzip *Nachrichten folgen Benutzer* (siehe Abschnitte 4.1 und 4.2). Netzmanager können auch beauftragt werden die Aufenthaltsorte mit Anzahl der gesendeten oder empfangenen Nachrichten der Mobilrechner zu beobachten und zu analysieren und gegebenenfalls Maßnahmen (u.a. durch eine Umkonfiguration) zu treffen. Diese Aufgaben können vereinfacht werden durch *dynamische User-Profiles* oder *dynamische Replikation von Nachrichten*.
   In der Regel sind Mailboxen nur an einem Ort positioniert und meistens hat jeder Benutzer eine Mailbox, das hat sich verändert mit der Einfhrung neuer Messaging-Architekturen, die die Anforderungen der Mobilität erfüllen sollen. Es soll weiter ermöglicht werden eine ganze Mailbox von einem Ort zu einem anderen zu verlagern. Die in Kapitel 4 vorgestellte Architektur definiert zwar verschiedene Schnittstellen, mit denen Mailboxen manipuliert werden, diese Möglichkeit besteht hier aber nicht. Eine mögliche Lösung wäre das Verpacken einer Mailbox in eine Nachricht, die dann weitergesckickt wird. Das ist dann eine Managementaufgabe. Information für

die Positionierung der Mailboxen und die Replikation der Nachrichten, die eventuel benötigt werden können, sind:

- Information über einzelne Mailboxen: Identifikator, Besitzer, Ort, Datei-Format und Größe.
- Information über einzelne Nachrichten. In diesem Fall ist der Wichtigkeitsgrad und der Besitzer einer Nachricht von Bedeutung.
- Der Name des Benutzers, seine Home-Adresse und eine alternative Adresse, die aus seiner Aufenthaltsinformation gewonnen wird, diese kann die *Care-of-adress* (Mobile-IP, vergleich Abschnitt 2.5) sein. Falls diese alternative Adresse nicht bestimmt werden kann, wird die Home-Adresse benutzt (Auf die Redundanz, die eintreten kann, und die zusätzlichen (SNMP)-Anfragen von der Managementanwendung, wird hier aufmerksam gemacht).
- Information über Verschiebungsziel einer Mailbox.
- Information über Verschiebungsziel einer Nachricht.

5. **Konfiguration von Filtern**
   Das automatische Verschicken von Nachrichten ist ein heutiges Merkmal von Messaging-Systemen, wie z.B. Elektronic-News. Das erfordert aber die *Konfiguration von Filtern* nach bestimmten Kriterien, wie der momentane Aufenthaltsort eines Benutzers (man denke an Wetter-Informationen), das Alter von Nachrichten oder auch *Prioritäten. Priorisierung von Nachrichten* für Filter-Zwecke stellt daher eine weitere Managementanforderung.
   Im nächsten Abschnitt wird auch gezeigt, wie Filter für Leistungszwecke nützlich sein können. Die benötigte Information ist u.a. :

   - Priorität, Wichtikeitsgrad, Größe, Ziel und Herkunft einer Nachricht.
   - Gesamtanzahl der Nachrichten in einer Mailbox.
   - Gesamtanzahl der Nachrichten eines Benutzers auch

6. **Group-Mailboxen**
   Ein weiteres Merkmal der heutigen Messaging-Protokolle sind Group-Mailboxen. Zugriffe auf eine Group-Mailbox (Vergleich Abschnitte 3.2.4 und 4.2.1) sind häufig und überlasten den Server, den Klient und das Übertragungsmedium. Durch eine geeignete *Konfiguration von Group-Mailboxen* könnte das Problem mindestens vermindert werden. Nachrichten sollen z.B. nach Wichtigkeit, Vertraulichkeitsgrad, Art oder Zugriffshäufigkeit klassifiziert werden und in periodischen Abständen werden einige an alle Benutzer in der Gruppe automatisch verschickt (ähnlich dem Broadcast-Verfahren, eine ausführliche Erklärung ist aus [IB95], [IVB94] zu entnehmen).
   Die Information, die benötigt wird:

- Die Art der Mailbox, sie kann eine Group- oder Privat-Mailbox sein.
- Die Zugriffshäufigkeit auf eine Nachricht.
- Wo die Mailbox ist und wie man sie identifizieren kann.

5.3 Leistungsmanagement

5.3.1 Einleitung

In einer mobilen Umgebung spielen die Verbindungen eine wichtige Rolle. Jede Verbindung sollte daher überwacht werden können mit gleichzeitiger Erstellung von Statistiken und Ausgabe von Informationen (Historie), sowie Feststellung der Kanalqualität. *Nach der in Kapitel 4 angegebenen Architektur* soll der Server mit den Komponenten MTA, MS, GW und AV der Mittelpunkt des Managements sein.

5.3.2 Anforderungen

1. **Massenspeicher- und Arbeitsspeichernutzung**
   Emailsysteme werden immer häufiger benutzt. Das hat zur Folge, daß die Anzahl der Nachrichten enorm gestiegen ist und damit auch die Speichernutzung. Messagingsysteme sind so konzipiert, daß ein hoher Zuverlässigkeitsgrad erreicht wird. D.h. Nachrichten und ihre Attachements sollen solange gespeichert werden, bis sie vom Benutzer gelöscht werden. In der in Kapitel 4 angegebenen Architektur werden Nachrichten auf dem Server gespeichert. Der Speicher wird von den Benutzern wenig oder gar nicht beachtet. Netzmanager sollen dann veraltete Nachrichten oder Mailboxen löschen und gegebenenfalls Mailboxen mit ihrem Inhalt komprimieren. Folgende Information wird gebraucht:

   - Wie die Nachrichten in der Mailbox gespeichert sind, d.h. komprimiert oder nicht.
   - Das Komprimierverfahren, des vom MS unterstützt wird.
   - **Das Alter** von Nachrichten.
   - Die zuletzt angekommene Nachricht in einer Mailbox und ihr Alter. Draus kann das Alter einer Mailbox gewonnen werden.
   - Die Information, ob eine Nachricht schon gelesen wurde oder nicht. Ungelesene Nachrichten dürfen nicht gelöscht werden.
   - Der Wichtigkeitsgrad und der Vertraulichkeitsgrad einer Nachricht, womit Grenzen für den Löschungsvorgang gesetzt werden können (siehe Sicherheitsmanagement).

2. **Kommunikationsvorgänge zwischen den Komponenten und den Übertragungsmedien**
   Die Qualität und die Eigenschaften der Übertragungsmedien sind sehr unterschiedlich (siehe [BBK94], [DBP94]). Dieses gilt für Durchsatz und Kosten. Das erfordert eine *Überwachung der Komponenten* und der *Einsatz von Cache-Verfahren*, um z.B. den wiederholten Transport gleicher Information zu verhindern. Geeignete Cache-verfahren können im Fall von einer Abkopplung und Wieder-Ankopplung eingesetzt werden, um einen *kontinuierlichen Betrieb zu gewährleisten*.
   Es soll ebenfalls möglich sein, das Verschicken von Nachrichten zu verzögern oder nur ein Teil einer Nachricht zu verschicken (**MIME-Parsing**). Ein Netz kann mit Nachrichten überflutet werden, *Nachrichten sollen gefiltert und systematisch klassifiziert* werden. Folgende Information wird dann benötigt:

   - Der Zeitpunkt, wann eine Verbindung aufgebaut wird, dieser soll veränderbar sein, um den Versand einer Nachricht zu verzögern.
   - Die Größe der Attachements von Nachrichten.
   - Die Bandbreite des Übertragungsmediums einer Verbindung.
   - Die Aktuell- und die Durschnittsgeschwindigkeit einer Verbindung.
   - Den Zeitpunkt des Abbruches einer Verbindung.
   - Die vorgesehene Zeit eine unterbrochne Verbindung wiederaufzubauen.
   - Der Grund, warum eine Verbindung unterbrochen wurde.
   - Alternativen zu einer momentanen Verbindung, d.h. ob eine Möglichkeit besteht das Übertragungsmedium zu wechseln. Das Verhältnis **Kosten-Leistung** soll hier berücksichtigt werden.

3. **Aktivitäten der Benutzer**
   Diese Anforderung ist stark mit den Verbindungen und der Aufenthaltsinformation des Benutzers verbunden und daher soll an dieser Stelle auf diesen angewiesen.

4. **Zustand des Servers z.B. Shutdown**
   Die Information, die diesen Punkt betrifft, ist aus den Network Services MIB ([RFC1565]) zu entnehmen.

5. Zustand der ausgetauschten Nachrichten (deliverable, undeliverable,...) Diese Anforderung betrifft MTAs. Lösungsansätze für MTAs gibt es bereits und werden erst im nächsten Kapitel behandelt.

6. **System-Unterstützung (UNIX, MINDOWS NT, NetWare,...)**
   Diese Anforderung betrifft zwar Messagingsysteme, Systemunterstützung ist

aber nicht messagingspezifisch und kann aus anderen MIBs ([?]) entnommen werden.

7. **Synchronisation und Update der Verzeichnisse**
   Diese Anforderung betrifft Adreßverzeichnisse und wird erst im nächsten Kapitel behandelt, weil es bereits Lösungsansätze gibt.

8. **Klient-Kapazitäten: CPU, Speicher und Energie**:
   Die *Optimierung der Reaktionszeiten* vom Server erspart Energie und die *Verlagerung der Funktionalität* auf der Server-Seite entlastet den Klient-CPU, das könnte aber Nachteile für das Übertragungsmedium, sowie für den Benutzer (steigende Kosten durch die längeren Verbindungszeiten) haben. Die benötigte Information ist u.a. :

   - Die Dauer einer Verbindung.
   - Die Kosten pro Minute der Benutzung eines Übertragungsmediums.
   - Die Eigenschaften des Klientrechners. In der in Kapitel 6 angegebenen MIB wird davon ausgegangen, daß die Kapazitäten des Klientrechners gering sind.

5.4 Sicherheitsmanagement

5.4.1 Einleitung

Mobilsysteme sind besonders gefährdet (siehe Abschnitt 2.4.1), weil die Kommunikationsmedien offen sind.
Die Messagingsystem-Komponenten sollen gegen unzulässige Zugriffe geschützt werden. Nachrichten und ihre Attachements sollen verschlüsselt sein. Es soll auch besonders Wert darauf gelegt werden, die Datenintegrität und eine gegenseitige Authentifizierung zu gewährleisten. Wie bereits in **Kapitel 2** gezeigt wurde, sind ausreichende Sicherheitsmechanismen auch mit Berücksichtigung der Schlüsselverteilmechanismen vorhanden.

5.4.2 Anforderungen

Im folgenden wird eine Zusammenfassung der Managementanforderungen gegeben und erläutert. Nicht alle Anforderungen werden in der MIB (siehe Kapitel 6) berücksichtigt, weil sie von exteren Anwendungen erfüllt werden sollen (Vergleich Abschnitt 2.4):

1. Physische Kontrolle über öffentliche und geheime Schlüssel zum *Schutz vor Fälschung eines Schlüssels*.

2. *Schutz vor Fälschung der Zeitangaben.* Diese kann zum Beispiel von finanziellem Interesse sein.

3. *Schutz vor Denial of Service*

4. *Physikalische Löschung von originellen, vertraulichen Daten.* Dateien werden von den meisten Betriebsystemen nicht gelöscht, sondern als gelöscht markiert, diese könnten dann gezielt gelesen werden.

5. Schutz vor Viren. Viren können den Klartext bei einer Verschlüsselung unbemerkt mithören und in andere Dateien schreiben, sie können auch den Verschlüsselungsvorgang beeinflussen, um gefälschte Angaben zu generieren.

6. *Schutz der Managementinformation selbst*, z.B knnten aus Statistiken, die über Verbindungen erstellt werden, wie der Nachrichtenherkunft, das Nachrichtenziel, die Nachrichtengröße vertrauliche Informationen gewonnen werden. Diese Anforderung soll durch das Management-Protokoll berücksichtigt werden (Vergleich Abschnitt 6.1).

7. *Passwort-Schutz*, Passwörter sollen nur fr denjenigen lesbar sein, dem sie gehren, sie sollen u.a. verschlüsselt gespeichert werden.

8. *Gespeicherte Nachrichten im Message Store sollen (müssen aber nicht) verschlüsselt* und unlesbar sein.

Diese Sicherheitsanforderungen werden zum Teil durch externe Sicherheitsmechanismen erfüllt . Einige sollen durch die Zugriffsrechte auf die MIB-Variablen erfüllt weden. Teile dieser Anforderungen werden in der MIB behandelt. Die benötigte Information ist:

- Der Name des Sicherheitsmechanismus, der untersützt wird. Mit IMAP-4 können PGP, Kerberos (IV oder V) oder ein einfaches Authentifizierungsverfahren (Login/Passwort) benutzt werden.

- Der Name des Sicherheitsalgorithmus (MD5, RSA, DES,...).

- Anzahl der Sicherheitsverletzungen (wird z.B. von IMAP-4 eingeloggt).

- Zeitpunkt der Sicherheitsverletzung.

- Der Grund einer Verletzung.

- Gebräuchlich ist ein Attribut, das gesetzt werden kann, um die Nachrichten in einer Mailbox zu verschlüsseln.

5.5 Fehlermanagement

5.5.1 Einleitung

Zuerst soll eine Zusammenfassung der **Randbedingungen** (siehe auch Abschnitt 1.2) , die das Fehlermanagement beeinflussen, angegeben werden. Die technischen Fehlerursachen, die mit dem Einsatz von Funkkanälen verbunden sind, sind u.a.:
Funkschatten
Tritt wegen der durch Abschaltung geringen Funkfeldstärke auf.
Intersymbol-Interferenz
Bedingt durch die schmale Bandbreite. Sie entsteht wegen Laufzeitunterschieden, die Reflexionen und Beugung verursachen, und aufgrund vonNachbarkanal-Störungen.
Systeminterne-Interferenz
Entsteht durch Störungen der Mobilfunkkanäle gleicher Frequenz von zwei Sendern.
Funklöcher
Werden verursacht durch Funkschatten und Schwundeffekte.
Schwunderscheinungen
Treten durch Signaleinbrüch oder Überlagerung von Signalen auf.
Dieses bedeutet u.a., daß die empfangenen Informationen von der gesendeten leicht abweichen könnten, das soll aber nicht als Fehler interpretiert werden. Eine weitere Tatsache in diesem Zusammenhang ist, daß die Abkopplung eines Mobilrechners vom Netz nicht bedeuten sollte, daß die Komponente ausgefallen ist.

5.5.2 Anforderungen

Die Fehlermanagementanforderungen sind:

1. **Der Nachrichtenfluß kontrollieren und überwachen**
   Durch z.B Message-Tracking, um Staus in Nachrichtenschlangen oder Loop-Zustände zu erkennen (siehe Abschnitt 4.2.1). Es soll ja ebenfalls ermöglicht werden, Meldungen von Benutzern z.B.über das nicht Ankommen einer Nachricht nachvollziehen zu können, und den Fehler im gesamten Weg einer Nachricht suchen zu können. Diese Anforderung betrifft zum Teil MTAs, daher wird hier auf Kapitel 6 verwiesen, in dem ein interessanter Ansatz von der Electronic Mail Association (EMA) vorliegt.
   Die, in der MIB (Message Store MIB) benötigte Information, ist dann:

   - Der Name des MTAs, von dem die Nachricht stammt.
   - Die Adresse des Agenten, der die MTA-MIB implementiert.
   - Das Ziel der Nachricht.

- Zustand einer Nachricht. Diese kann zerstört sein und daher unlieferbar.
- Grund der Zerstörung einer Nachricht (z.B. durch ein Virus).
- Ein Attribut, das gesetzt werden kann, um eine Nachricht zu reparieren, falls eine Möglichkeit besteht.

2. **Überwachen der Verbindungen**
Um u.a. mobil-bedingte Störungen, die oben genannt wurden, zu erkennen, sollen Verbindungen überwacht werden. Die Information die benötigt wird, ist:

   - Der Grund des Abbruches einer Verbindung (siehe auch Leistungsmanagement)
   - Der Grund der Ablehnung eines Verbindungswunsches (siehe auch Sicherheitsmanagement), nach innen und außen.
   - Der Zeitpunkt des Geschehens eines Verbindungsfehlers.
   - Die Kanalqualität, diese kann aus der Kosten, Durchsatz und Bandbreite interpretiert werden (siehe Leistungsmanagement).
   - Anzahl der fehlenden und der abgebrochenen Verbindungen.

3. **Analyse und Filtern von Fehlermeldungen**
Die Analyse von Fehlermeldungen wird von der Managementanwendung vorgenommen. Filter für die Fehler werden in der MIB nicht berücksichtigt.

4. **Analyse von unterbrochenen Verbindungen**
Dieses umfaßt u.a. die Erkennung von abgekoppelten Zuständen und die Fortsetzung der Verbindung (Restart oder ab dem Punkt der Unterbrechung). Bei IMAP-4 wird dieses Verhalten empfohlen und bleibt implementierungsabhängig. Zwei Attribute, das eine zum Fortsetzen einer Verbindung nach dem Zeitpunkt der Unterbrechung und das zweite zum Neustarten einer Verbindung, werden benötigt. Diese sollen aber optional sein.

5.6 Abrechnungsmanagement

Für das Abrechnungsmanagement sind individuelle Informationen über Benutzer, einzelne Nachrichten und die Kommunikationsvorgänge durch *Message Tracking vom Sender bis zum Empfänger* notwendig. *Filterkriterien*, wie die Größe, das Sendedatum, der Sender, der Empfänger, das Subjekt, ..., sollen definiert werden. Managementansätze für diesen Zweck gibt es bereits (siehe Kapitel 6). *Eine Nachricht soll über ein Managementdomain hinaus verfolgt werden können.* Das ist

auch für das Fehlermanagement wichtig, da dadurch Rückschlüsse auf die Nichtfunktionstüchtigkeit einer Messaging-Komponente gewonnen werden können. Ausführlich wird das in den vier anderen Managementfunktionen (Abschnitt 5.2-5) behandelt.
Messaging gewinnt heute immer mehr an kommerzieller Bedeutung, daher sollen die *Kosten erfaßt* und *Statistiken erstellt werden*. Die dafür benötigte Information ist:

- Anfangs- und Beendigungszeit einer Verbindung.
- Die Grösse der ausgetauschten Datenmengen, hier werden Nachrichten (Kopf und Rumpf) betrachtet.

Ressourcenverwaltung
Mobilrechner können dynamisch zwischen verschiedenen Bereichen (siehe **Abschnitt 4.1**) wandern, dadurch ergeben sich Änderungen in der Gesamtstruktur des Systems. Das ergibt die Notwendigkeit einer *dynamischen Verwaltung der Systemkonfiguration* (betrachte z.B *file-Zugriff*).
Diese Anforderung ist zwar für Messagingsysteme interessant, sie ist aber nicht spezifisch. Deshalb wird sie in der MIB nicht berücksichtigt.
Benutzerverwaltung:
Zentrale Benutzerverwaltung, wie das Eintragen, Löschen oder die Änderung von Benutzern und ihren Rechten und die Einrichtung von Gruppen durchzuführen. Benutzer werden mit Mailboxen verbunden. Die Einrichtung eines neuen Benutzers bzw. einer Gruppe von Benutzern bedeutet die Einrichtung einer Privat- bzw. Group-Mailbox. Folgende Attribute werden benötigt:

- Ein Attribut zum Löschen aller Mailboxen eines Benutzers.
- Ein Attribut zum Löschen und einer zum Kreieren Mailbox.
- Ein Attribut zum Sperren eines Benutzers aus einer Group-Mailbox.
- Ein Attribut zum Vergeben von Schreibrechten in einer Group-Mailbox für einen Benutzer.

5.7 Zusammenfassung

Die Managementanforderungen, die in diesem Kapitel aufgezeigt wurden, können in vier verschiedene Kategorien untergliedert werden:

1. Anforderungen, die in der MIB aufgenommen werden,
2. Anforderungen, die technisch zu realisieren sind,
3. Anforderungen, die nicht messagingspezifisch sind,

4. und Anforderungen, die in von existierenden Managementansätzen erfüllt werden.

Eine Zusammenfassung, der bis zu diesem Kapitel behandelten Themen, ist in der folgenden Abbildung dargestellt.

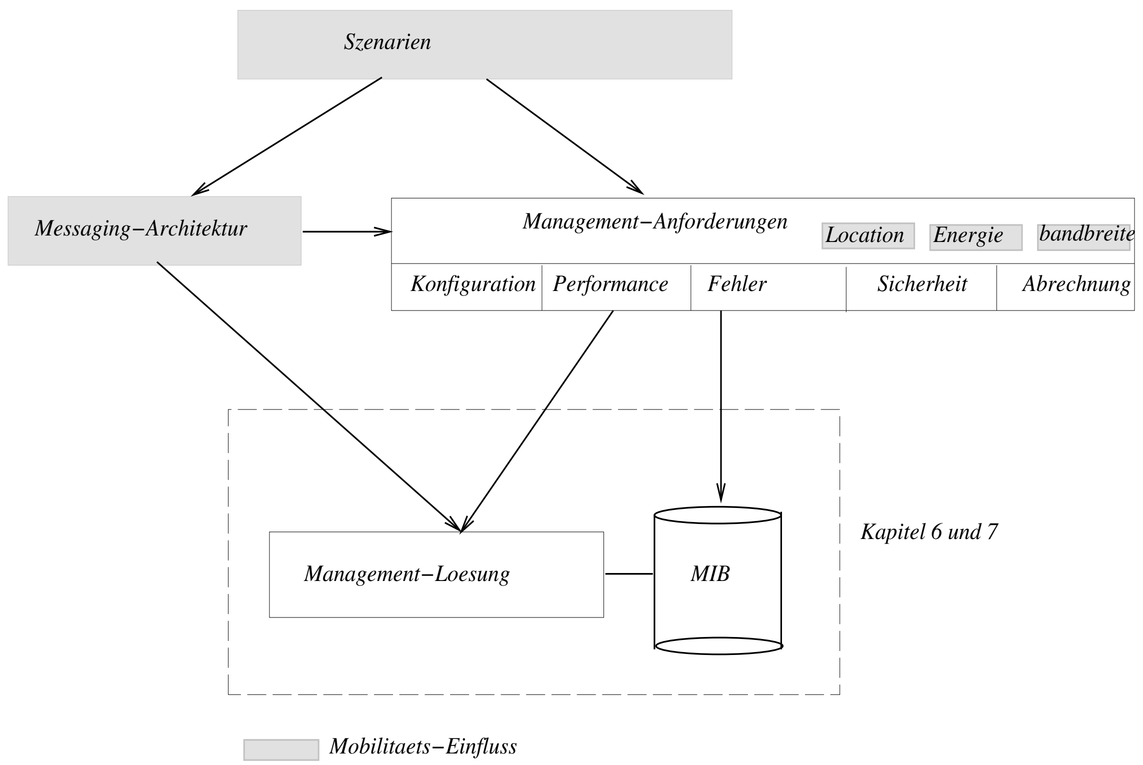


Abbildung 5.1: Zusammenfassung

Kapitel 6

Internet-Management eines Messagingsystems

In Kapitel 5 wurden die Managementanforderungen entlang der Management-Funktionsbereiche aufgezählt. In diesem Kapitel werden in einem ersten Schritt existierende Managementansätze vorgestellt. In einem weiteren Schritt wird aus den Mängel dieser Ansätzen eine Management Information Base (MIB) für den Message Store angegeben.

6.1 Existierende Ansätze

6.1.1 Message Tracking

Den Nachrichtenfluß (siehe Abschnitt 4.2) zu überwachen, Information über einzelne Benutzer (Sender und Empfänger) festzulegen ist, wie in **Kapitel 5** gezeigt wurde, sind wichtige Managementanforderungen. Bis zum jetzigen Zeitpunkt ist das in den Internet-Standards (im Bezug auf das Management von Messagingsystemen) unbeachtet geblieben. Die EMA ([EMA95a]) hat eine Message-Tracking-MIB definiert, die in den folgenden Abschnitten vorgestellt wird.

- **Nachrichten-Historie**
  Es werden Informationen über die Historie von einzelnen Nachrichten gesammelt und gespeichert, das kann z.B. benutzt werden, um zu überprüfen ob die Messaging-Komponenten korrekt funktionieren.

- **Momentane Nachrichtenstatus mit dem gesamten Nachrichtenverlauf**
  Eine Nachricht kann einer oder mehrere Empfänger haben, die Überwachung und gegebenenfalls die Optimierung der Nachrichtenverlauf für jeden Empfänger und mit bestimmten Filterkriterien ist in der MIB enthalten.

- **EMA-Empfehlungen**
  Erwähnungswert in dieser Arbeit von der EMA ist u.a.:

  1. Die Agenten der Messagingkomponenten sollen über eine gewisse Intelligenz verfügen um z.B. einige Operationen automatisch durchzuführen.
  2. Automatische *follow-up*-Abfragen von der Management-Station an den Agenten.
  3. Nicht nur Identifikatoren zur eindeutigen Identifizierung der Nachrichten benutzen sondern auch andere Attribute, es gibt einige Messagingprotokolle, die Message IDs nicht unterstützen.

Message-Tracking-MIB

1. *mtaInformationTable*
   Information über MTA's deren Agenten abgefragt werden sollen: der Name der MTA, das darunterliegende Messaging-Protokoll, die Zeit der letzten verfügbaren Information über diesen MTA, die Adresse eines alternativen Agenten, der alte Informationen bereitstellen kann.

2. *mtaMessageTable*
   Diese Tabelle enthält sowohl Query-Attribute (zum Filtern) als auch Response-Attibute. Es enthält Information über einzelne Nachrichten, Empfänger und Sender und insbesondere Filterkriterien : der Nachricht-Identifikator, der Sender, das Subjekt, die minimale Größe, die maximale Größe, ...

3. *gatewayMessageTable*
   Information über Nachrichten, Empfänger und Gateways von Nachrichten, die konvertiert werden sollen weil ein andere Messaging-Protokoll benutzt werden soll um die Nachricht weiterzuleiten: z.B die Proritát oder die Größe.

6.1.2 Network Services MIB

Die Network Services MIB ([RFC1565]) ist für das Monitoring von Anwendungsserver spezifiziert worden. Sie ist so aufgebaut, daß sie als allgemeingültig (in bezug auf Anwendungsserver) bezeichnet werden kann.
Monitoring-Information von speziellen Anwendungen soll in neudefinierten, dazugehörigen MIB's enthalten. Die Revision von der EMA ([EMA95a]) enthält keine Vorschläge zur Erweiterung oder Veränderung dieser MIB bis auf zwei Attributen, die in den folgenden Abschnitten erwähnt werden.

1. *applTable*
   Jeder Eintrag in der Anwendungs-Tabelle hat 15 Attribute die folgende Information enthalten sollen:

Spezifische Information über die Anwendung: ein Index um eine lexikographische Ordnung in der Tabelle zu ermöglichen, ein Name, mit dem die Anwendung angesprochen werden kann, der Name des Verzeichnisses, wo statische Information über die Anwendung gespeichert ist, die Version der Anwendung, die Zeit seit der letzten Initialisierung der Management-Software, der momentane Anwendungsstatus und die Zeit der letzten Änderung einer Anwendung.
Der Anwendungsstatus wurde von der EMA um den Wert *quiescing* erweitert, was bedeutet, daß die Anwendung im Prozeß zum *down*-Zustand ist. Aus dem *appUpTime* kann einige wichtige Information über die operationale Dauer einer Anwendung durch Anwendung der folgenden Regel gewonnen werden:
sysUpTime ([RFC1213]) − appUpTime = Operationale Dauer einer Anwendung
Information über die Verbindungen einer Anwendung: der momentane Anzahl der hinein- und herausgehende Verbindungen einer Anwendung, die Zeit seit der letzten Verbindung nach außen und nach innen, die Zahl der abgelehnten Verbindungswünsche und die Anzahl der fehlenden Verbindungen nach außen.

2. *assocTable*
   Allgemeine Information über einzelne Verbindungen: ein Index zur eindeutigen Identifizierung einer Verbindung, Name der Remote-Anwendung, Typ der für diese Anwendung benutzte Protokoll, Art der Remote-Anwendung (z.B : UA) und die Dauer der Verbindung.

6.1.3 Mail Monitoring MIB

Diese MIB ([RFC1566]) ist konzipiert für das Monitoring von MTA's, sie hat zwei Tabelle : eine Tabelle für statische Information über MTA's und eine zweite für die Überwachung der Verbindungen, der Nachrichtenverlauf und der Nachrichtenschlangen, diese wurden von der EMA ([EMA95a]) um Alarme, die im Falle von MTA- oder Nachrichtenfehler generiert werden, erweitert.

1. *mtaTable*
   Spezifische, statische Information über MTA's: wie z.B Anzahl, Größe und Empfänger-Anzahl der empfangenen, gesendeten und momentan gespeicherten Nachrichten.

2. *mtaGroupTable*
   Jede Zeile enthält Informationen über den Nachrichtenverlauf zwischen ein MTA und eine andere Messaging-Komponente, in Abbildung 6.1 wird das veranschaulicht:

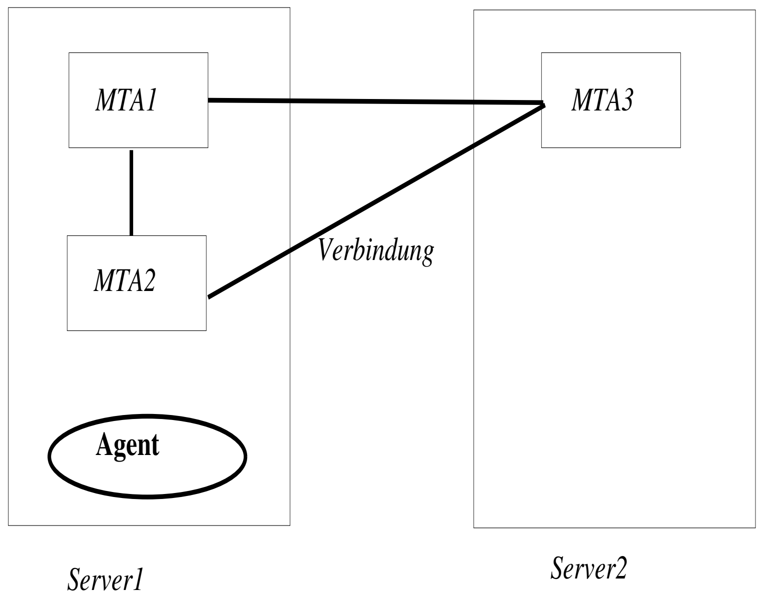


Abbildung 6.1: Verbindungen einer mtaGroup

Für den Agent im Server1 enthält die mtaGroupTable vier Einträge: MTA1-MTA2, MTA1-MTA3, MTA2-MTA1, MTA2-MTA3. Die mtaTable enthält nur zwei Einträge für MTA1 und MTA2.
Die Attribute dieser Tabelle sind zusätzlich zu denen von *mtaTable* und *assocTable*: die älteste Nachricht, die beim MTA gespeichert ist, Grund der letzten Ablehnung einer Nachricht, Grund der letzten fehlenden Verbindung, die Verzögerungszeit bis zur nächsten Verbindungsaufbau.

3. *Erweiterungen der EMA*
Die EMA hat zusätzliche Attribute, die besonders für Gateways wichtig sind definiert, diese sind die Anzahl der konvertierten Nachrichten und die Anzahl der fehlenden Konvertierungen und der Nachrichten Identifikator der ältesten Nachricht.
Die EMA hat diese MIB auch um Alarme, die in der Management-Station ausgelöst werden sollen, erweitert.

6.1.4 X.500 Directory Monitoring MIB

Das X.500 Verzeichnis ist für heterogene Umgebungen besonders wichtig (siehe **Kapitel 2**). Adressen und Information über Benutzern können dort einheitlich gepflegt oder von dort abgefragt werden.
Diese MIB [RFC1567]) enthält drei Tabellen:

1. *dsaOpsTable*
Statische Information über Zugriffe auf den DSA, Operationen und Fehler.

2. *dsaEntriesTable*
   Information über DSA-Einträge und Cache-Information.

3. *dsaIntTable*
   Information über Interaktionen mit anderen DSAs.

6.2 Management des Message Stores

6.2.1 Managementfunktionen

Es werden folgende Managementfunktionen (siehe **Kapitel 5**) unterstützt:

1. Überwachung der Verbindungen sowohl mit MTAs als auch mit Klienten.

2. Erstellung von spezifischen Statistiken über den Message Store und die Verbindungen

3. Verwaltung der Mailboxen und ihren Besitzern

4. Message-Tracking: Nachrichten werden in ihrem gesamten Lebenszyklus überwacht und gegebenenfalls zurückverfolgt

5. Information über Benutzer und Benutzersitzungen bereitstellen

6. Speicherverwaltung

7. Es werden Sicherheitsaspekte berücksichtigt.

6.2.2 Informationsmodell

Das Informationsmodell stellt das Kernstück einer Managementlösung, die Managementinformation wird aus den Managementanforderungen (Siehe Kapitel 5) und entlang des Objektmodells (siehe Kapitel 4) generiert. Die Darstellung der Managementinformation unterscheidet sich nach dem unterliegenden Informationsmodell, führend in diesem Bereich sind das OSI- und das IAB-Modell (Internet) ([HA93]):

1. **OSI-Modell**
   Die Management-Information wird objektorientiert modelliert wobei folgende Eigenschaften unterstützt werden:

   - Kapselung: Attribute eines Objektes werden nur durch seine Methoden manipuliert.
   - Vererbung: Objekte mit ähnlichen Eigenschaften werden in einer Klasse zusammengefaßt, Klassen haben eine Hierarchie, Objekte einer Klasse erben die Attribute und Methoden einer Oberklasse.

2. **IAB-Modell**
   Die Management-Information wird durch einfache Variablen modelliert, die in einem Registrierungsbaum angehängt werden. Dieses Modell unterstützt weder Vererbung noch Kapselung und ist durch Informationsredundanz gekennzeichnet. Das IAB-Modell ist aber einfach und kommt am meisten zum Einsatz besonders im kleinen und mittleren Bereich.

Im folgenden wird das IAB-Modell angewendet. Die Objekte werden durch Tabellen, deren Zeilen die Attribute des Objekts repräsentieren, definiert. Komplexe Attribute, die durch sogenannten Nebentabellen definiert werden sollen, werden auf einfache konfigurierbare Attribute abgebildet. Die Syntax fr die Definition der Managed Objects (MOs) wird ASN.1 sein.

6.2.3 Message Store Tabellen

Diese MIB wird fünf Tabellen enthalten:

1. *msTable*
   Statische Information über den Message Store.

2. *folderTable*
   Information über einzelne Mailboxen und deren Eigentümer.

3. *connectionTable*
   Information über einzelne Verbindungen.

4. *messageTable*: Information über Nachrichten.

5. *userTable*: Information über Benutzer.

6.2.4 MessageStore-MIB

- *msTable*
  Diese Tabelle enthält Information über den Message Store. Diese Information wird Hauptsäschlich aus Abschnitt 5.2.2 abgeleitet.

```
msTable OBJCT-TYPE
        SYNTAX SEQUENCE OF msEntry

msEntry ::= SEQUENCE {
                           msIndex Integer32,
                           msName DisplayString,
                           msService Integer,
                           msPortNumber Integer32,
                           msVersion DisplayString,
                           msMaximumLoginTrials Integer32,
```

```
                           msEnvelopLookahead Integer32,
                           msUIDLookahead Integer32,
                           msNewsSpoolDirectory DisplayString,
                           msStorageVolume Gauge32,
                           msStorageFormat Integer,
                           msFolderMaxNumber Integer32 ,                                                                             msCurrentFolderNumber Integer32 ,
                           msPrivFolderMaxSize Gauge32 ,
                           msGruopFolderMaxSize Gauge32 ,
                           msReceivedMessages Counter32 ,
                           msReceivedVolume Gauge32 ,
                           msStoredMessages Counter32 ,
                           msReceivedMessages Gauge32 ,
                           msReceivedRecipients Counter32 ,
                           msInboundConnections Integer32 ,
                           msOutboundConnections Integer32 ,
                           msRejectedInboundConnections Counter32 ,
                           msAbortedInboundConnections Counter32 ,
                           msSecurityAlgorithm DisplayString,
                           msSecurityViolationNumber Counter32
}
```

– **msIndex**
  Ein Index zur eindeutigen Identifizierung dieses MSs.

```
msName OBJECT-TYPE
  SYNTAX Integer32
  MAX-ACCESS read-only
  DESCRIPTION
               "A unique identifier for this ms "
```

– **msName**
  Ein Name, mit dem der Message Store angesprochen werden kann.

```
msName OBJECT-TYPE
  SYNTAX DisplayString
  MAX-ACCESS read-only
  DESCRIPTION
               " A string representation for the
                  name of the message store"
```

– **msService**
  Ein Integer, der den Namen der Service dieses Message Stores repräsentiert

```
msService OBJECT-TYPE
  SYNTAX Integer
  MAX-ACCESS read-only
  DESCRIPTION
             " An Integer containing the name
                of the sevice for this message store:
                (1): imap4
                (2): imap2bis
                (3): rfc1176 (imap2)
                (4): pop3
                (5): other"
```

- **msPortNumber**
  Die Port-Nummer für diesen Service z.B für imap4: 143.

```
msPortNumber OBJECT-TYPE
  SYNTAX Integer32
  MAX-ACCESS read-only
  DESCRIPTION
            " An integer schowing the port number
               for this MS"
```

- **msVersion**
  Die Version der Implementierung dieses MS, verschiedene Implementierungen des gleichen Protokolles können unterschiedlich oder erweitert sein, für Fehlerkorrekturen ist es von besonderen Wichtigkeit die Version abfragen zu können.

```
msVersion OBJECT-TYPE
  SYNTAX DisplayString
  MAX-ACCESS read-only
  DESCRIPTION
              " A string representation for the
                Version of this message store"
```

Die folgenden vier Attribute können in einigen Message Stores nicht festgelegt werden, in diesem Fall kann der Wert auf Null gesetzt werden. Der Grund ist, daß sie nicht bei allen Protokollen definiert sind, oder in einigen Fällen nicht implementiert sind.

- **msMaximumLoginTrials**
  Die maximale Anzahl der Loginversuche, die der Klient durchführen kann, bevor er abgewiesen wird. Diese Zahl kann erhöht werden bei niedrigem

Netzverkehr.

```
msMaximumLoginTrials OBJECT-TYPE
  SYNTAX Integer32
  MAX-ACCESS read-write
  DESCRIPTION
              " An integer schowing the maxium of
                iterations allowed in the login state"
```

– **msEnvelopLookahead**
Die Anzahl der Nachrichtenumschläge, die von einem Klient automatisch geholt werden können nach z.B einem Suchkommando, diese werden (müssen) dann vom Klient zwischengespeichert werden um die Anzahl der Transaktionen zu minimieren oder zu maximieren wenn ein Klient nicht in der Lage ist zwischenzuspeichern (Cache) .

```
msEnvelopLookahead OBJECT-TYPE
  SYNTAX Integer32
  MAX-ACCESS read-write
  DESCRIPTION
               " An integer schowing the number
                  of envelops wich are automatically
                  fetched in a kommand (e.g search).
                  When not available the value is -1"
```

– **msUIDLookahead**
Die Anzahl der eindeutigen Nachrichten Identifikatoren, die nach einem Kommando durchsucht werden (siehe voriges Attribut).

```
msUIDLookahead OBJECT-TYPE
  SYNTAX Integer32
  MAX-ACCESS read-write
  DESCRIPTION
              "An integer schowing the number of uids
               that are looked ahead in a UID-Kommand.
               When not available the value is zero"
```

– **msNewsSpoolDirectory**
Einige Messaging-Protokolle unterstützen auch NEWS z.B USENET NEWS. Dieses Attribut bezeichnet die Name der Spool-Verzeichnis der NEWS.

```
msNewsSpoolDirectory OBJECT-TYPE
```

```
    SYNTAX DisplayString
    MAX-ACCESS read-write
    DESCRIPTON
                        " A String representation for the locatin of the
                           NEWS spool of  this MS.
                           When not availible this string is empty"
```

- **msStorageVolume**
  Der Speicher-Volumen, der für diesen MS reserviert ist.

```
  msStorageVolume OBJECT-TYPE
    SYNTAX Gauge32
   UNIT "Byte"
    MAX-ACCESS read-write
    DESCRIPTION
               " The amount of storage volume
                  required for this ms measered in bytes"
```

- **msStorageFormat**
  Dieser Attribut wird benötigt um komprimierten von nicht komprimierten
  Daten zu unterscheiden .

```
  msStorageFormat OBJECT-TYPE
    SYNTAX Integer
    MAX-ACCESS read-only
    DESCRIPTION
                " The storage format for this ms.
                   configuration: compressed and encrypted (1),
                                  not compressed and encrypted (2),
                                  compressed and not encrypted (3),
                                  not compressed and not encrypted (4)"
```

- **msFolderMaxNumber**
  Der Maximal-Anzahl der Mailboxen in dem Message Store, diese Anzahl
  könnte erhöht werden wenn die Grösse der Mailboxen klein bleibt.

```
  msFolderMaxNumber OBJECT-TYPE
    SYNTAX Integer32
    MAX-ACCESS read-write
    DESCRIPTION
               " An integer containing the max_number
                  of folder in the message store"
```

– **msCurrentFolderNumber**
Momentane Anzahl der Mailboxen.

```
msCurrentFolderNumber OBJECT-TYPE
  SYNTAX Integer32
  MAX-ACCESS read-only
  DESCRIPTION
               " An Integer containing the current
                  number of folder in the message store"
```

– **msPrivFolderMaxSize**
Die maximale Größe einer privaten Mailbox in dem Message Store, diese Anzahl könnte manipuliert werden wenn diese Größe zu hoch ist. der Mailboxen klein bleibt.

```
msPrivFolderMaxSize OBJECT-TYPE
  SYNTAX Gauge32
  UNIT "Bytes"
  MAX-ACCESS Read-Write
  DESCRIPTION
                " An integer containing the maxsize
                  of private folder in the message store.
                  measered in bytes"
```

– **msGroupFolderMaxSize**
Die maximale Größe eine Group-Mailbox. Sie wird auf null gesetzt falls, keine Group-Mailboxen unterstützt werden.

```
msGroupFolderMaxSize OBJECT-TYPE
  SYNTAX Gauge32
  UNIT "Bytes"
  MAX-ACCESS read-write
  DESCRIPTION
            " An Integer containing the maxsize
               of Group folder in the message store. measered
               in bytes. zero (0) means group folder are not
               present"
```

Folgende Attribute sind für statistische Zwecke gedacht:

– **msReceivedMessages**
Die Gesamt-Anzahl der Nachrichten, die der MS seit seine Initialisierung

empfangen hat.

```
msReceivedMessages OBJECT-TYPE
 SYNTAX Counter32
 MAX-ACCESS read-only
 DESCRIPTION
      "The number of messages received
       since ms initialization."
```

– **msReceivedVolume**
Der Speicher-Volumen aller Nachrichten, die der MS empfangen hat seit seiner Initialisierung .

```
msReceivedVolume OBJECT-TYPE
    SYNTAX Gauge32
    UNITS "Bytes"
    MAX-ACCESS read-only
    DESCRIPTION
      "The total volume of messages received
        since ms initialization, measured in bytes ."
```

– **msStoredMessages**
Die Anzahl der momentan gespeicherten Nachrichten.

```
msStoredMessages OBJECT-TYPE
    SYNTAX Integer32
    MAX-ACCESS read-only
    DESCRIPTION
      "The total number of messages
        currently stored in the ms."
```

– **msStoredVolume**
Der Speicher-Volumen der momentan im MS gespeicherten Nachrichten.

```
msStoredVolume OBJECT-TYPE
  SYNTAX Gauge32
  UNITS "Bytes"
  MAX-ACCESS read-only
  DESCRIPTION
    "The total volume of messages currently
     stored in the ms, measured in bytes ."
```

– **msReceivedRecipients**
Der Anzahl der Empfänger der von dem MS empfangenen Nachrichten.

```
msReceivedRecipients OBJECT-TYPE
   SYNTAX Counter32
   MAX-ACCESS read-only
   DESCRIPTION
     "The total number of recipients
      specified in all messages
      received since ms initialization."
```

Statischtische Information über Verbindungen

– **msInboundConnections**
Anzahl der momentanen Verbindungen zum MS, z.B mit MTA's wenn eine Nachricht kommen soll oder mit Klients, die die Nachrichten Abfragen wollen.

```
 msInboundConnections OBJECT-TYPE
   SYNTAX Integer32
   MAX-ACCESS read-only
   DESCRIPTION
     "The number of current connections to the ms,
       where the ms is the responder."
```

– **msOutboundConnections**
Die Anzahl Verbindungen vom ms z.b wenn neue Mail ankommt und eine Information an einem Klienten weitergeleitet soll.

```
msOutboundConnections OBJECT-TYPE
   SYNTAX Integer32
   MAX-ACCESS read-only
   DESCRIPTION
     "The number of current Connections to
      the ms, where the ms is the initiator."
```

– **msRejectedInboundConnections**
Anzahl der abgelehnten Verbindungen z.B wegen einen Authentifizierungsfehler.

```
msRejectedInboundConnections OBJECT-TYPE
   SYNTAX Counter32
   MAX-ACCESS read-only
   DESCRIPTION
     "The total number of inbound
      connections the ms has
      rejected, since ms initialization."
```

- **msAbortedInboundConnections**
  Anzahl der abgebrochenen Verbindungen z.B wegen einer schlechten Kanal-Qualität.

```
msAbortedInboundConnections OBJECT-TYPE
   SYNTAX Counter32
   MAX-ACCESS read-only
   DESCRIPTION
     "The total number of aborted inbound
      connection, since ms initialization."
```

- **msSecurityAlgorithmName**
  Name des Sicherheitsmechanismus, das von diesem MS unterstützt wird.

```
msSecurityAlgorithm OBJECT-TYPE
   SYNTAX Integer
   MAX-ACCESS read-only
   DESCRIPTION
     "The name of the security algorithm
       used in this ms. Configuration: PEM (1)
                                       PGP (2)
                                       MD5 (3)
                                       Kerberos IV (4)
                                       Kerberos V (5)
                                       Login/password (6)
                                       Other  (7) "
```

- **msSecurityViolationNumber**
  Die Gesamtanzahl der Sicherheitsverletzungen.

```
msSecurityViolationNumber OBJECT-TYPE
   SYNTAX Counter32
   MAX-ACCESS read-only
   DESCRIPTION
     "The total number of security violation
       since ms initialisation"
```

  Sicherheitsverletzungen werden hier nicht im einzelnen behandelt. Falls eine Notwendig besteht kann das ergänzt werden.

- *folderTable*

```
folderTable OBJCT-TYPE
        SYNTAX SEQUENCE OF folderEntry

folderEntry ::= SEQUENCE {
                           folderID DisplayString,
```

```
                              folderOwner Integer32,
                              folderType Integer,
                              folderCreationDate DateandTime,
                              folderFormat DisplayString,
                              folderDirectory DisplayString,
                              folderStoredMessages Integer32,
                              folderNewestMessageID DisplayString,
                              folderMostAccessedMessageID DisplayString,
                              folderAction Integer
}
```

– **folderID**
  Ein Identifikator um die Mailbox eindeutig zu identifizieren.

```
itemize   folderID OBJECT-TYPE
       SYNTAX DisplayString
       MAX-ACCESS read-only
       DESCRIPTION
         "The string representation for the
          folder ID"
```

– **folderOwner**
  Ein Identifikator für den Besitzer (userID) oder die Besitzer (groupID) dieser Mailbox.

```
   folderID OBJECT-TYPE
       SYNTAX Integer32
       MAX-ACCESS read-write
       DESCRIPTION
         "An Integer schowing the identifier of the
           owner (userID or groupID) this folder"
```

– **folderType**
  Der Typ der Mailbox: eine Private Mailbox, eine Group-Mailbox, eine NEWS-Mailbox oder eine Fremd-Mailbox.

```
   folderType OBJECT-TYPE
       SYNTAX Integer
       MAX-ACCESS read-only
       DESCRIPTION
         "The string representation for the
          folder type.
          configuration: private (1),
                         group (2),
                         news (3),
                         foreign (4)"
```

– **folderCreationDate**
  Die Zeit des Kreirens der Mailbox.

```
folderCreationDate OBJECT-TYPE
    SYNTAX DateandTime
    MAX-ACCESS read-only
    DESCRIPTION
      "Time that the folder was created"
```

- **folderFormat**
  Das Dateiformat, das benutzt wird um Emails in der Mailbox zu speichern
  .

```
folderFormat OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-only
    DESCRIPTION
      "A string representation for the format
        In wich data is written into mailbox"
```

- **folderDirectory**
  Das Verzeichnis, wo sich diese Mailbox befindet.

```
folderDirectory OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-only
    DESCRIPTION
      "A string representation for the directory
        in which the folder is located "
```

- **folderStoredMessages**
  Die Anzahl der momentan in der Mailbox gespeicherten Nachrichten.

```
folderStoredMessages OBJECT-TYPE
    SYNTAX Integer32
    MAX-ACCESS read-only
    DESCRIPTION
      "An integer containing the current
       number of messages stored in the
       folder"
```

- **folderStoredMessagesVolume**

```
folderStoredMessageVolume OBJECT-TYPE
    SYNTAX Gauge32
    UNITS "Bytes"
    MAX-ACCESS read-only
    DESCRIPTION
      "The total volume of messages currently
       stored in the folder, measured in bytes"
```

– **folderNewstMessageID**
Der Nachrichten-Identifikator der neusten Nachricht in dieser Mailbox. Weitere Information über diese Nachricht kann dann aus der messageTable entnommen werden.

```
folderNewstMessageID OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-only
    DESCRIPTION
      "An Integer containing the newst
       message ID in this folder"
```

– **folderMostAccessedMessageID**
Der Identifikator der Nachricht, auf der am meisten zugegriffen wurde in dieser Mailbox, diese könnte dem Klient oder den Klienten zugeschickt werden und dann gelöscht.

```
folderMostAccessedMessageID OBJECT-TYPE
    SYNTAX Integer32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
      "An integer containing the most accessed
        Message ID in this folder"
```

– **folderAction**
Mit diesem Attribut können Mailboxen manipuliert werden.

```
folderAction OBJECT-TYPE
    SYNTAX Integer
    MAX-ACCESS read-write
    DESCRIPTION
      "This an action field. Possible values are:
        1 = add a folder
        2 = delete a folder
        3 = send the folder as a message to the
              folderOwner"
```

• *messageTable*

```
messageTable OBJCT-TYPE
        SYNTAX SEQUENCE OF messageEntry

messageEntry ::= SEQUENCE {
                           messageID DisplayString,
                           messageFolderID DisplayString,
                           messageEncoding Integer,
```

```
                              messageArrivalTime DateandTime,
                              messageCreationDate DateandTime,
                              messagePriority DisplayString,
                              messageSize Gauge32,
                              messageMTASourceName DisplayString,
                              messageMTASourceType DisplayString,
                              messageMTASourceAgent DisplayString,
                              messageSupplementalInformation DisplayString,
                              messageAccess Counter32,
                              messageAction Integer
}
```

– **messageID**
Ein eindeutiger Identifikator für jede Nachricht, Nachrichten können bei einigen Mailsysteme nicht eindeutig identifizierbar sein. In diesem Fall sollte dieser Identifikator der Nachricht zugewiesen werden (siehe Kapitel 5).

```
MessageID OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-write
    DESCRIPTION
               "A string representation for the unique identifier
                 for this message"
```

– **messageFolderID**
Der Identifikator der Mailbox, in der die Nachricht gespeichert ist.

```
messageFolderID OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-only
    DESCRIPTION
              " A string representation for the ID of
                 of the folder in wich this message is stored"
```

– **messageEncoding**
Die Codierung dieser Nachricht.

```
messageEncoding OBJECT-TYPE
    SYNTAX Integer
    MAX-ACCESS read-only
    DESCRIPTION
      "An integer schowing the message body encoding,
        the configuration is :
        (1): 7 bit SMTP semantic data
        (2): 8 bit SMTP semantic data
        (3): 8 bit binary data
        (4): base-64 encoded data
        (5): human-readable 8-as-7 bit data
```

```
        (6): other
        (7): unknown"
```

- **messageArrivalTime**
  Die Zeit, wann die Nachricht zum Message Store angekommen ist.

```
messageArrivalTime OBJECT-TYPE
    SYNTAX DateandTime
    MAX-ACCESS read-only
    DESCRIPTION
      "Time that the this message arrived
        at this ms"
```

- **messageCreationDate**
  Die Zeit, wann diese Nachricht dem ersten MTA übermittelt wurde.

```
MessageCreationDate OBJECT-TYPE
    SYNTAX DateandTime
    MAX-ACCESS read-only
    DESCRIPTION
      "Time that the message was submitted
         to the first MTA"
```

- **MessageSize**
  Die Grösse diese Nachricht. Die Ubermittlung von Nachrichten, die zu groß sind kann beim Hochverkehr verzögert werden .

```
MessageSize OBJECT-TYPE
    SYNTAX Gauge32
    UNIT "Bytes"
    MAX-ACCESS read-only
    DESCRIPTION
      "Size in bytes of this Message"
```

- **MessagePriority**
  Die Priorität dieser Nachricht, spielt eine große Rolle bei der Verzögerung der Zustellung einige Nachrichten und auch bei der Entscheidung ob eine alte Nachricht gelöscht werden soll.

```
MessagePriority OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-only
    DESCRIPTION
      "A string reprasentation for the priority
        of this message"
```

Die Nächsten drei Attribute sind von der Message Tracking MIB insperiert und bereiten die Möglichkeit, Nachrichten zurückzuverfolgen vor

– **messageMTASourceName**
Die Name des MTAs, von dem der MS die Nachricht bekommen hat .

```
messageMTASourceName OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-only
    DESCRIPTION
      "A string reprasentation for the name of the
        mta from wich the MS received the Message"
```

– **messageMTASourceType**
Die Typ des MTAs (z.B SMTP), von dem der MS die Nachricht bekommen hat .

```
messageMTASourceType OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-only
    DESCRIPTION
      "A string reprasentation for the type of the
        mta from wich the MS received the Message"
```

– **messageMTASourceAgent**
Die Adresse des Agenten des MTAs, von dem der MS die Nachricht bekommen hat .

```
messageMTASourceAgent OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-only
    DESCRIPTION
      "A string reprasentation for the adress of the
       agent for the MTA appearing in the
       messageMTASource attribute"
```

– **messageSupplementalInformation**
Zusätzliche Information über diese Nachricht, die frei definiert werden könnte.

```
messageSupplementalInformation OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-write
    DESCRIPTION
      "A string reprasentation for a supplemental
       infomation to this Message"
```

– **messageAccess**
Ein Zähler für die Zugriffe auf diese Nachricht.

```
messageAccess OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
```

```
        DESCRIPTION
          "How many this message was accessed"
```

– **messageAction**
  Dieses Attribut kann gesetzt werden um eine Nachricht zu manipulieren.

```
    messageAction OBJECT-TYPE
        SYNTAX Integer
        MAX-ACCESS read-write
        DESCRIPTION
          "This is an action field. Possible values are:
            1 = delete the message
            2 = compress the message
            3 = encrypt the mesaage
            4 = send the message to the user specified
                  In the to field
            5 = send the message to the adress specified
                   in the userHomeMailAdress Attribute
            6 = send the message to the adress specified
                   in the UserForeignMailAdress Attribute "
```

In diese Tabelle können noch folgende Attribute zugefügt werden, falls es notwendig ist (siehe Kapitel 5.6.2). Diese sind: Kosten pro Minute einer Verbindung und die Größe der ausgetauschten Datenmengen (Nachrichten: Kopf und Rumpf).

- *connectionTable*

```
connectionTable OBJCT-TYPE
        SYNTAX SEQUENCE OF connectionEntry

connectionEntry ::= SEQUENCE {
                           connectionIndex Integer32,
                           connectionAverageSpeed Integer32,
                           connectionMechanism DisplayString,
                           connectionUpTime Integer32,
                           connectionClientHost DisplayString,
                           connectionInboundAbortReason DisplayString,
                           connectionInboundFailureReason DisplayString,
                           connectionRestartTime DateandTime,
                           connectionAbortTime DateandTime,
                           connectionAction Integer
}
```

– **connectionIndex**
  Ein Index zur eindeutigen Identifizierung einer Verbindung.

```
connectionIndex OBJECT-TYPE
   SYNTAX Integer32
   MAX-ACCESS read-only
   DESCRIPTION
     "An index to uniquely identify each
      connection for this ms"
```

- **connectionAverageSpeed**
  Die Durschnittsgeschwindigkeit der Verbindung zwischen der Remote-Anwendung (UA, MTA) und der Message Store.

```
connectionAverageSpeed OBJECT-TYPE
   SYNTAX Integer32
   MAX-ACCESS read-only
   DESCRIPTION
     " A string showing the connection
       speed between the client or
       mta and the ms measered in bytes/second"
```

- **connectionMechanism**
  Der benutzte Verbindungsmechanismus, z.B *netwareconnect*

```
 connectionMechanism OBJECT-TYPE
   SYNTAX DisplayString
   MAX-ACCESS read-only
   DESCRIPTION
     " A string showing the connection
       mechanism beiing used"
```

- **connectionUptime**
  Die Dauer der Verbindung bis zum jetzigen Zeitpunkt.

```
connectionUpTime OBJECT-TYPE
    SYNTAX Integer32
    MAX-ACCESS read-only
    DESCRIPTION
      "An integer containing the current
       connection time in seconds"
```

- **connectionClientHost**
  Name des Rechner des Klienten.

```
connectionRemAppType OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-only
    DESCRIPTION
      "A string showing the name of the
        client host"
```

- **connectionInboundAbortReason**
  Der Grund des Abbruchs dieser Verbindung, falls sie abgebrochen wurde. Ein leerer String bedeutet, daß die Verbindung erfolgreich war.

```
connectionInboundAbortReason OBJECT-TYPE
   SYNTAX DisplayString
   MAX-ACCESS read-only
   DESCRIPTION
     "The abort reason.
      An empty string indicates that
      the last attempt was successful. "
```

- **connectionInboundRestartTime**
  Die Zeit des Wiederaufsetzens, wenn die Verbindung abgebrochen wurde. Ein Null-Wert bedeutet, daß die Verbindung erfolgreich war.

```
connectionInboundRestartTime OBJECT-TYPE
   SYNTAX DateandTime
   MAX-ACCESS read-write
   DESCRIPTION
     "The time when this connection
       was aborted.
       a zero value indicates
       the last attempt was successful. "
```

- **connectionInboundAbortTime**
  Die Zeit, wann dieser Verbindung abgebrochen wurde. Ein Null-Wert bedeutet, daß die Verbindung erfolgreich war.

```
connectionInboundAbortTime OBJECT-TYPE
   SYNTAX DateandTime
   MAX-ACCESS read-only
   DESCRIPTION
     "The time when this connection was aborted.
      a zero value indicates
     the last attempt was successful. "
```

- **connectionInboundFailureReason**
  Der Grund der Fehler einer Verbindung.

```
connectionInboundAbortReason OBJECT-TYPE
   SYNTAX DisplayString
   MAX-ACCESS read-only
   DESCRIPTION
     "The failure reason.
      An empty string indicates that
      the last attempt was successful."
```

– **connectionAction**
  Ein Attribut, der gesetzt werden kann, um eine Verbindung zu unterbrechen oder wieder aufzubauen .

```
connectionAction OBJECT-TYPE
   SYNTAX Integer
   MAX-ACCESS read-only
   DESCRIPTION
     "This is an action field. Possible values are:
       1 = abort this connection
       2 = restart this connection"
```

- *userTable*
  In Abschnitt 5.6.2 wurde auf die Notwendigkeit der Benutzerverwaltung aufmerksam gemacht. Diese Tabelle ist für die Benutzerverwaltung.

```
userTable OBJCT-TYPE
        SYNTAX SEQUENCE OF userEntry

userEntry ::= SEQUENCE {
                           userIndex Integer32,
                           userFolderList DisplayString,
                           userAction Integer,
                           userHomeMailAdress DisplayString,
                           userForeignMailAdress DisplayString
}
```

– **userIndex**
  Ein Identifikator für den Benutzer, der den BenutzerID entspricht.

```
userIndex OBJECT-TYPE
    SYNTAX Integer32
    MAX-ACCESS read-only
    DESCRIPTION
               "The index for the user which is
                 equal to the userID"
```

– **userFolderList**
  Eine Liste der Identifikatoren der Mailboxen, die diesem Benutzer gehören und der Groupmailboxen, in denen dieser Benutzer registriert ist.

```
userFolderList OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-write
    DESCRIPTION
               "A sequence of strings with the identifiers of
                 all folders for this user include group folder"
```

- **userAction**
  Ein Identifikator für den Benutzer, der den BenutzerID entspricht.

```
userAction OBJECT-TYPE
    SYNTAX Integer
    MAX-ACCESS read-write
    DESCRIPTION
               "This is the action field. Possible values are
                 1 = add a user
                 2 = delete a user
                 3 = subscribe to a group folder
                 4 = unsubscribe from a group folder
                 5 = read-write in a group folder"
```

- **userHomeMailAdress**
  Ein Home-Adresse des Benutzer.

```
userHomeMailAdress OBJECT-TYPE
    SYNTAX IpAdress
    MAX-ACCESS read-only
    DESCRIPTION
               "The mail adress of this user"
```

- **userForeignMailAdress**
  Die in den Managementanforderungen definierte alternative Adresse.

```
userForeignMailAdress OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-write
    DESCRIPTION
               "The foreign mail adress of this user
                 (e.g username@care-of-adress)"
```

Kapitel 7

Implementierung

In diesem Kapitel wird zuerst eine Managementarchitektur dargestellt und anschließend ein konkreter Implementierungsvorschlag basierend auf den, im Rahmen des Projektes *SNMP-basiertes Systemmanagement* ([HK95]), vorhandenen Arbeiten, angegeben.

7.1 Simple Network Management Protocol (SNMP)

SNMP ([SNMP]) ist ein einfaches Management-Protokoll, das in einem breiten Umfeld eingesetzt wird und einfach handhabbar ist. Die Eigenschaften von SNMP lassen sich nach den zwei Versionen SNMPv1 und SNMPv2 einordnen:

- **SNMPv1**
  SNMPv1 ist ein einfaches Protokoll zur Austausch der Managementinformation, die SNMP-Agenten senden Ereignisse, Pilling bernimmt der Manager. SNMPv1 unterstüzt nicht das *Scoping und Filterung*-Verfahren wegen den fehlenden objekt-orientierte Informationsmodell, das Protokoll ist ebenfalls durch Sicherheits- und Leistungsmängel gekennzeichnet ([HA93]).

- **SNMPv2**
  SNMPv2 ist eine Erweiterung der ersten Version, die wichtigsten Merkmale sind:

  1. Unterstützung von Subagenten Registrierung (Varbind Multiplexing)
  2. Ermöglichen der Manager-To-Manager Kommunikation
  3. Mehr Sicherheit (Report PDU) und Leistung (Beschleunigung der Suche in Tabellen durch GetBulk PDU)
  4. Austauschmöglichkeit von Anwendungsspezifischen Daten.
  5. Reduktion der SNMP-Transaktionen (RMON, RMON/2)

6. Unterstützung von Highspeed-Netzwerke z.B ATM durch die Einführung von 32 und 64-bit Datentypen.
7. Die Möglichkeit die Kapazitätsgrenzen eines Agenten zu beschreiben.

7.2 Organisationsmodell

Die Lokalisierung und Spezifikation der Komponenten eines Messagingsystems lassen eine Managementarchitektur basierend auf SNMP angeben, die in Abbildung 7.1 skizziert ist.

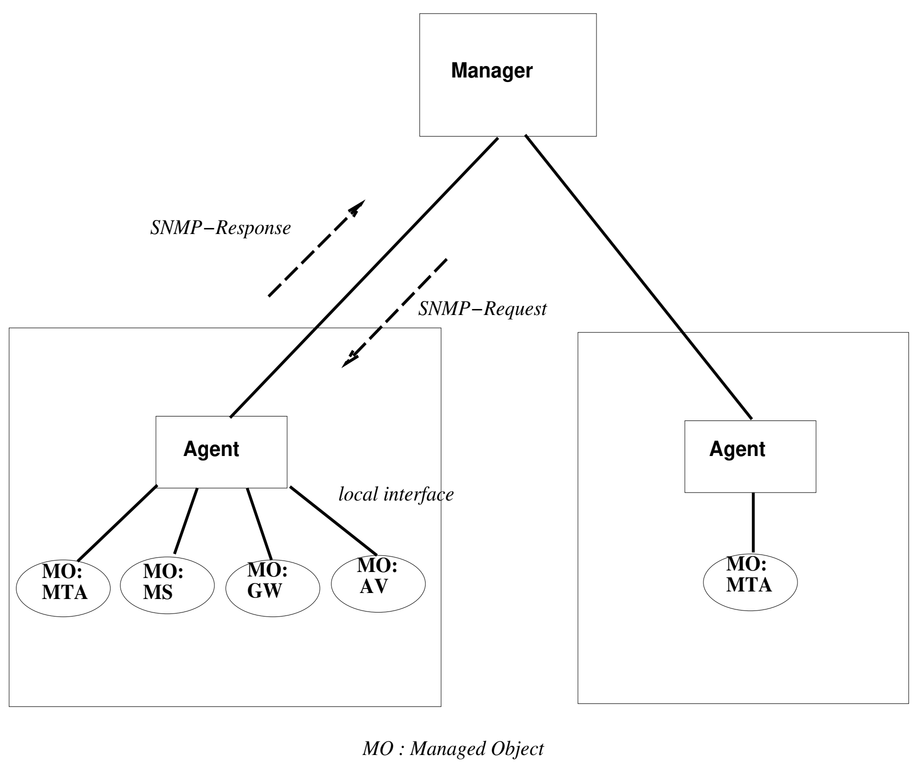


Abbildung 7.1: Managementarchtitektur eines Messagingsystems

Diese Anwendung interagiert mit dem Agenten um Informationen über die Komponente des Massagingsystems abzufragen. Der Manager hat keine spezifische Kenntnisse über die Implementierung der Messaging-Komponente, deshalb kommuniziert er mit einem Agenten, der lokal bei der Messaging-Komponente residiert und über eine lokale Schnittstelle die Information über die Komponente holt.

7.3 Implementierung mit DPI-Subagenten

Die im Rahmen dieses Projektes ([HK95]) realisierten Komponenten (siehe Abbildung 7.2) sind ein SNMPv1/v2-fähige Hauptagent, der die MIB-II implementiert. Er wurde um eine Logging-Komponente und eine DPI-Server-Schnittstelle erweitert, die für die Kommunikation mit den Subagenten dient. Die Subagenten implementieren diverse spezifische MIBs und verfügen ebenfalls über DPI-Schnittstellen.

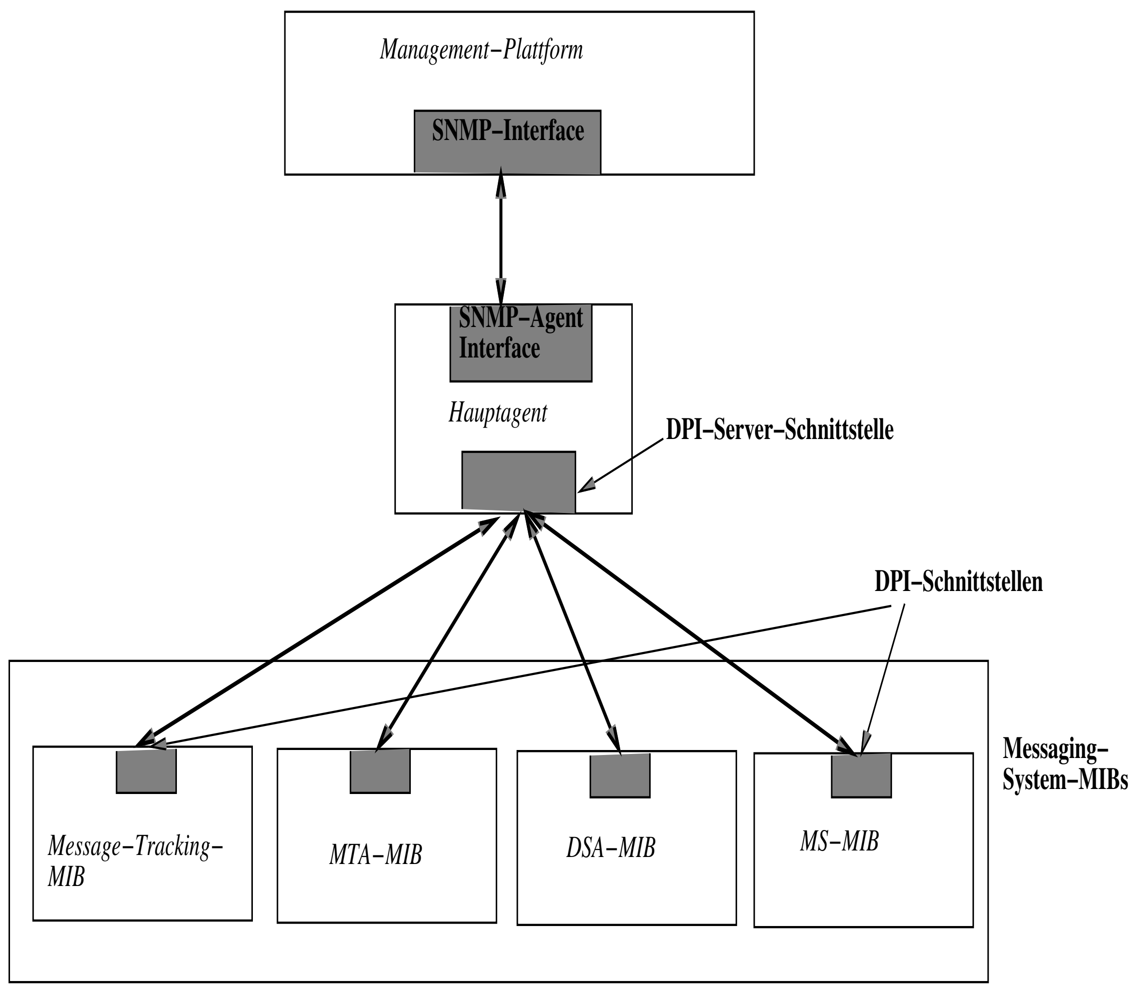


Abbildung 7.2: Management-Architektur eines Messagingsystems

7.3.1 Erweiterung um Managementkomponenten für Messagingsysteme

Wie bereits in Kapitel 6 demonstriert wurde, sind für das Management eines Messagingsystems diverse spezifische MIBs (MTA-Monitoring-MIB, Message-Tracking-

MIB, DSA-MIB, MS-MIB) notwendig, diese können im Rahmen des Projektes durch Plattformübergreifende Agenten implementiert werden, da ein Email-System meistens nicht auf ein spezifisches System festgelegt ist.

MS-MIB Implementierung

Diese MIB kann im MS-Domain (z.B imapd) implementiert werden, dieser soll dann um eine DPI-Schnittstelle erweitert werden, dadurch vereinfacht sich die Kommunikation zwischen der Management Komponente und der zu managenden MS-Domain. Nachteile sind: es kann auf einem einzigen Rechner verschiedene MS-Domaine gleichzeitig laufen, die dann alle um Management-Schnittstellen erweitert werden sollen. Der Absturz einer MS-Domain bedeutet auch der Absturz der zu Management-Komponente, was nicht vernünftig erscheinet für ein Email-System.
Die zweite Möglichkeit ist ein unabhängiger Agent, der verschiedene Domaine, die auf einem Rechner laufen zuständig ist, dadurch lassen sich die Nachteile des ersten Ansatzes vermeiden. Der Implementierungsaufwand ist dann allerdings größer und ein zusätzliche Kommunikationsprozeß Agent-MS-Domain ist auch zu realisieren.

Kapitel 8

Zusammenfassung und Ausblick

Dieses abschließende Kapitel soll ein Rückblick auf die Arbeit und ein Ausblick auf Zukünftige Entwicklungsmöglichkeiten sein.

8.1 Ziele der Arbeit

Die wesentlichen Ziele dieser Arbeit bestanden darin:

- Die Untersuchung des Einflußes von Mobilität auf Messagingprotokolle
- Die Definition einer für das mobile Umfeld geeigneten Messagingarchitektur
- Die Spezifikation der Komponenten dieser Architektur
- Die Ableitung der Managementanforderungen
- Die Untersuchung der existierenden Managementansätze
- Die Wahl einer zu managenden Komponente und die Ableitung der dafür benötigte Managementinformation
- Die Aufstellung einer MIB für das Management der ausgewählten Komponente

8.2 Zusammenfassung

In der vorliegenden Diplomarbeit wurden folgende Themen behandelt:

- **Kapitel 1:**
  Es wurden Beispiele für den Einsatz von Messagingsystemen aus der Praxis nach unterschiedlichen Kriterien (Heterogenität, Sharing) aufgezeigt.

- **Kapitel 2:**
  Ausgewählte Messagingprotokolle und -architekturen, Sicherheitstechniken und die Mobilitätstechniken wurden vorgestellt.

- **Kapitel 3:**
  Die technische Einsatzszenarien von Messagingsystemen wurden im Hinblick auf die Mobilität der Endgeräte näher betrachtet, analysiert und miteinander verglichen.

- **Kapitel 4:**
  Eine für das mobile Umfeld geeignete Messagingarchitektur (IMAP-4 mit SMTP) wurde um weitere Komponenten wie z.B. X.500 ergänzt. Die Einzelnen Komponenten und ihre Funktionalität wurden beschrieben. Anschließend wurde ein Objektmodell für den Message Store angegeben.

- **Kapitel 5:**
  Hier wurden die Managementanforderungen entlang der Managementfunktionsbereichen abgeleitet (aus der in Kapitel 4 angegebenen Architektur) und daraus die Managementinformation abstrahiert.

- **Kapitel 6:**
  Im ersten Teil wurden existierende standardisierte und nicht standardisierte MIBs vorgestellt. Es stellte sich heraus, daß der Message Store unbeachtet blieb. Der zweite Teil wurde dann für die Angabe der Managementinformation (MIB), die für das Management des Message Stores benötigt wird, reserviert.

- **Kapitel 7:**
  Ein Implementierungskonzept wurde kurz angegeben.

- **Kapitel 8:**
  Dieses lesen Sie gerade.

8.3 Zeitrahmen der Arbeit

Die Zeit, die zur Realisierung dieser Arbeit aufgewendet wurde, kann mit etwa *9 Monaten* angesetzt werden. Darin eingeschlossen sind:

- Die Einarbeitung in den Messagingarchitekturen und -protokolle sowie die Sicherheitstechniken und Mobilitätstechniken (überproportional großer Einarbeitungsaufwend).

- Die Durchführung der oben genannten Ziele

- Die vorliegende schriftliche Ausarbeitung

8.4 Ausblick

Messagingsysteme werden zunehmend in Datennetze integriert. Gateways, die verschiedene Messagingsysteme, sowie Messagingdienste verbinden, gewinnen immer an Bedeutung und bilden den Schlüssel für die Integration, sie stellen auch den Übergang zu mobilen Umgebungen. Bis zum heutigen Zeitpunkt wurde diese Komponente als ein spezielles MTA betrachtet, bis auf einige Betrachtungen der EMA. Weitere Schritte, die diese Diplomarbeit folgen können sind:

1. Entwicklung einer eigenständigen MIB für Gateways unter Berücksichtigung, daß diese zu unterschiedlichen Zwecken eingesetzt werden können.

2. Weiteruntersuchung der Protokollarchitekturen auf Integrationsmöglichkeiten (Beispiel: SNMP-IMAP4)

3. Implementierung der spezifischen MIBs eines Messagingsystems.

Anhang A

Definition der Managed Objects für den Message Store

```
-- MS-MIB
-- Diplomarbeit an der TU-Muenchen
-- Aufgabensteller: Prof. Dr. H.-G.Hegering
-- Autor: Salhi Mohsen
-- e-mail: salhi@nm.informatik.uni-muenchen.de

MS-MIB DEFINITIONS ::= BEGIN

IMPORTS
       MODULE-IDENTITY, OBJECT-TYPE,
       Integer32, Integer, Counter32, Gauge32
              FROM SNMPv2-SMI
       DisplayString, DateandTime
              FROM SNMPv2-TC
      mib-2 FROM RFC1213-MIB

ms MODULE-IDENTITY
     LAST-APDATED "96........... "
     ORGANISATION " Munich Network Management Team"
     CONTACT-INFO
               " Salhi Mohsen
                  E-Mail: salhi@nm.informatik.uni-muenchen.de"
     DESCRIPTION
               " The MIB module describing Message Stores (MSs)"

::= {mib-2 .... }

-- msTable
-- A Table holding information about Message Stores
msTable OBJCT-TYPE
        SYNTAX SEQUENCE OF MsEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
```

```
             " The table holding information specific to an ms"
::= { ms 1 }

msEntry OBJCT-TYPE
        SYNTAX MsEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
             " An entry associated with each ms "
::= {maTable 1}

MsEntry ::= SEQUENCE {
                           msIndex Integer32,
                           msName DisplayString,
                           msService Integer,
                           msPortNumber Integer32,
                           msVersion DisplayString,
                           msMaximumLoginTrials Integer32,
                           msEnvelopLookahead Integer32,
                           msUIDLookahead Integer32,
                           msNewsSpoolDirectory DisplayString,
                           msStorageVolume Gauge32,
                           msStorageFormat Integer,
                           msFolderMaxNumber Integer32 ,
                           msCurrentFolderNumber Integer32 ,
                           msPrivFolderMaxSize Gauge32 ,
                           msGruopFolderMaxSize Gauge32 ,
                           msReceivedMessages Counter32 ,          msReceivedVolume Gauge32 ,          msStoredMessages Counter32 ,
                           msReceivedMessages Gauge32 ,
                           msReceivedRecipients Counter32 ,
                           msInboundConnections Integer32 ,
                           msOutboundConnections Integer32 ,
                           msRejectedInboundConnections Counter32 ,
                           msAbortedInboundConnections Counter32 ,
                           msSecurityAlgorithm DisplayString,
                           msSecurityViolationNumber Counter32
}

  msIndex OBJECT-TYPE
       SYNTAX Index
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
                    " A unique identifier for this ms"
    ::= { msEntry 1}

   msName OBJECT-TYPE
       SYNTAX DisplayString
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
```

```
                " A string representation for the
                   name of the message store"
::= { msEntry 2}

msService OBJECT-TYPE
   SYNTAX Integer
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
              " An Integer containing the name
                 of the sevice for this message store:
                 (1): imap4
                 (2): imap2bis
                 (3): rfc1176 (imap2)
                 (4): pop3
                 (5): other"
 ::= {msEntry 3}

 msPortNumber OBJECT-TYPE
   SYNTAX Integer32
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
             " An integer schowing the port number
                for this MS"
 ::= {msEntry 4}

 msPortNumber OBJECT-TYPE
   SYNTAX Integer32
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
             " An integer schowing the port number
                for this MS"
 ::= {msEntry 5}

 msMaximumLoginTrials OBJECT-TYPE
   SYNTAX Integer32
   MAX-ACCESS read-write
   STATUS current
   DESCRIPTION
               " An integer schowing the maxium of
                 iterations allowed in the login state"
 ::= {msEntry 6}

 msEnvelopLookahead OBJECT-TYPE
   SYNTAX Integer32
   MAX-ACCESS read-write
   STATUS current
   DESCRIPTION
                " An integer schowing the number
```

```
                  of envelops wich are automatically
                  fetched in a kommand (e.g search).
                  When not available the value is -1"
::= {msEntry 7}

msUIDLookahead OBJECT-TYPE
  SYNTAX Integer32
  MAX-ACCESS read-write
  STATUS current
  DESCRIPTION
              "An integer schowing the number of uids
               that are looked ahead in a UID-Kommand.
               When not available the value is zero"
::= {msEntry 8}

msNewsSpoolDirectory OBJECT-TYPE
  SYNTAX DisplayString
  MAX-ACCESS read-write
   STATUS current
   DESCRIPTON
                          " A String representation for the locatin of the
                             NEWS spool of  this MS.
                             When not availible this string is empty"
::= {msEntry 9}

msStorageVolume OBJECT-TYPE
  SYNTAX Gauge32
  UNIT "Bytes"
  MAX-ACCESS read-write
  STATUS current
  DESCRIPTION
             " The amount of storage volume
                required for this ms measered in bytes"
::= {msEntry 10}

msStorageFormat OBJECT-TYPE
  SYNTAX Integer
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
      " The storage format for this ms.
        configuration:
              compressed and encrypted (1),
              not compressed and encrypted (2),
              compressed and not encrypted (3),
              not compressed and not encrypted (4)"
::= {msEntry 11}

msFolderMaxNumber OBJECT-TYPE
  SYNTAX Integer32
  MAX-ACCESS read-write
```

```
  STATUS current
  DESCRIPTION
          " An integer containing the max number
            of folder in the message store"
::= {msEntry 12}

msCurrentFolderNumber OBJECT-TYPE
  SYNTAX Integer32
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
          " An Integer containing the current
            number of folder in the message store"
::= {msEntry 13}

msPrivFolderMaxSize OBJECT-TYPE
  SYNTAX Gauge32
  UNIT "Bytes"
  MAX-ACCESS Read-Write
  STATUS current
  DESCRIPTION
          " An integer containing the maxsize
            of private folder in the message store.
            measered in bytes"
::= {msEntry 14}

msGroupFolderMaxSize OBJECT-TYPE
  SYNTAX Gauge32
  MAX-ACCESS read-write
  UNIT "Bytes"
  STATUS current
  DESCRIPTION
         " An Integer containing the maxsize
           of Group folder in the message store. measered
           in bytes. zero (0) means group folder are not
           present"
::= {msEntry 15}

 msReceivedMessages OBJECT-TYPE
  SYNTAX Counter32
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
       "The number of messages received
        since ms initialization."
::= {msEntry 16}

msReceivedVolume OBJECT-TYPE
  SYNTAX Gauge32
  UNITS "Bytes"
  MAX-ACCESS read-only
```

```
   STATUS current
   DESCRIPTION
     "The total volume of messages received
       since ms initialization, measured in bytes ."
 ::= {msEntry 17}

 msStoredMessages OBJECT-TYPE
   SYNTAX Integer32
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
     "The total number of messages
       currently stored in the ms."
 ::= {msEntry 18}

 msStoredVolume OBJECT-TYPE
   SYNTAX Gauge32
   UNITS "Bytes"
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
     "The total volume of messages currently
      stored in the ms, measured in bytes ."
::= {msEntry 19}

msReceivedRecipients OBJECT-TYPE
   SYNTAX Counter32
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
     "The total number of recipients
      specified in all messages
      received since ms initialization."
 ::= {msEntry 20}

 msInboundConnections OBJECT-TYPE
   SYNTAX Integer32
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
     "The number of current connections to the ms,
       where the ms is the responder."
 ::= {msEntry 21}

 msOutboundConnections OBJECT-TYPE
   SYNTAX Integer32
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
     "The number of current Connections to
      the ms, where the ms is the initiator."
```

```
    ::= {msEntry 22}

    msRejectedInboundConnections OBJECT-TYPE
      SYNTAX Counter32
      MAX-ACCESS read-only
      STATUS current
      DESCRIPTION
        "The total number of inbound
         connections the ms has
         rejected, since ms initialization."
    ::= {msEntry 23}

    msAbortedInboundConnections OBJECT-TYPE
      SYNTAX Counter32
      MAX-ACCESS read-only
      STATUS current
      DESCRIPTION
        "The total number of aborted inbound
         connection, since ms initialization."
    ::= {msEntry 24}

    msSecurityAlgorithm OBJECT-TYPE
      SYNTAX Integer
      MAX-ACCESS read-only
      STATUS current
      DESCRIPTION
        "The name of the security algorithm
          used in this ms. Configuration: PEM (1)
                                          PGP (2)
                                          MD5 (3)
                                          Kerberos IV (4)
                                          Kerberos V (5)
                                          Login/password (6)
                                          Others (7) "
    ::= {msEntry 25}

    msSecurityViolationNumber OBJECT-TYPE
      SYNTAX Counter32
      MAX-ACCESS read-only
      STATUS current
      DESCRIPTION
        "The total number of security violation
          since ms initialisation"
    ::= {msEntry 26}

-- folderTable
-- A table holding information about folders

folderTable OBJCT-TYPE
        SYNTAX SEQUENCE OF FolderEntry
        MAX-ACCESS not-accessible
        STATUS current
```

```
        DESCRIPTION
             " The table holding information specific to a folder"
::= {ms 2}

folderEntry OBJCT-TYPE
        SYNTAX FolderEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
             " An entry associated with each folder "
::= {folderTable 1}

FolderEntry ::= SEQUENCE {
                           folderID DisplayString,
                           folderOwner Integer32,
                           folderType Integer,
                           folderCreationDate DateandTime,
                           folderFormat DisplayString,
                           folderDirectory DisplayString,
                           folderStoredMessages Integer32,
                           folderNewestMessageID DisplayString,
                           folderMostAccessedMessageID DisplayString,
                           folderAction Integer
}

    folderID OBJECT-TYPE
       SYNTAX DisplayString
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
         "The string representation for the
          folder ID"
    ::= {folderEntry 1}

    folderOwner OBJECT-TYPE
       SYNTAX Integer32
       MAX-ACCESS read-write
       STATUS current
       DESCRIPTION
         "An Integer schowing the identifier of the
           owner (userID or groupID) this folder"
     ::= {folderEntry 2}

     folderType OBJECT-TYPE
       SYNTAX Integer
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
         "The string representation for the
          folder type.
          configuration: private (1),
```

```
                    group (2),
                    news (3),
                    foreign (4)"
::= {folderEntry 3}

folderCreationDate OBJECT-TYPE
  SYNTAX DateandTime
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
    "Time that the folder was created"
::= {folderEntry 4}

folderFormat OBJECT-TYPE
  SYNTAX DisplayString
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
    "A string representation for the format
      In wich data is written into mailbox"
::= {folderEntry 5}

folderDirectory OBJECT-TYPE
  SYNTAX DisplayString
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
    "A string representation for the directory
      in which the folder is located "
::= {folderEntry 6}

folderStoredMessages OBJECT-TYPE
  SYNTAX Integer32
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
    "An integer containing the current
     number of messages stored in the
     folder"
::= {folderEntry 7}

folderStoredMessageVolume OBJECT-TYPE
  SYNTAX Gauge32
  UNITS "Bytes"
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
    "The total volume of messages currently
     stored in the folder, measured in bytes"
::= {folderEntry 8}
```

```
     folderNewstMessageID OBJECT-TYPE
       SYNTAX DisplayString
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
         "An Integer containing the newst
          message ID in this folder"
     ::= {folderEntry 9}

     folderMostAccessedMessageID OBJECT-TYPE
       SYNTAX Integer32
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
         "An integer containing the most accessed
           Message ID in this folder"
     ::= {folderEntry 10}

     folderAction OBJECT-TYPE
       SYNTAX Integer
       MAX-ACCESS read-write
       STATUS current
       DESCRIPTION
         "This an action field. Possible values are:
           1 = add a folder
           2 = delete a folder
           3 = send the folder as a message to the
                 folderOwner"
     ::= {folderEntry 11}

-- messageTable
-- A table holding information about messages

messageTable OBJCT-TYPE
        SYNTAX SEQUENCE OF MessageEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
             " The table holding information specific to a message"
::= { ms 3 }

messageEntry OBJCT-TYPE
        SYNTAX MessageEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
             " An entry associated with each message "
::= { messageTable 1 }

MessageEntry ::= SEQUENCE {
                           messageID DisplayString,
                           messageFolderID DisplayString,
```

```
                          messageEncoding Integer,
                          messageArrivalTime DateandTime,
                          messageCreationDate DateandTime,
                          messagePriority DisplayString,
                          messageSize Gauge32,
                          messageMTASourceName DisplayString,
                          messageMTASourceType DisplayString,
                          messageMTASourceAgent DisplayString,
                          messageSupplementalInformation DisplayString,
                          messageAccess Counter32,
                          messageAction Integer
}

   messageID OBJECT-TYPE
       SYNTAX DisplayString
       MAX-ACCESS read-write
       STATUS current
       DESCRIPTION
                  "A string representation for the unique identifier
                    for this message"
   ::= {messageEntry 1}

   messageFolderID OBJECT-TYPE
       SYNTAX DisplayString
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
                 " A string representation for the ID of
                    of the folder in wich this message is stored"
   ::= {messageEntry 2}

   messageEncoding OBJECT-TYPE
       SYNTAX Integer
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
         "An integer schowing the message body encoding,
           the configuration is :
           (1): 7 bit SMTP semantic data
           (2): 8 bit SMTP semantic data
           (3): 8 bit binary data
           (4): base-64 encoded data
           (5): human-readable 8-as-7 bit data
           (6): others
           (7): unknown"
   ::= {messageEntry 3}

   messageArrivalTime OBJECT-TYPE
       SYNTAX DateandTime
       MAX-ACCESS read-only
       STATUS current
```

```
    DESCRIPTION
      "Time that the this message arrived
        at this ms"
 ::= {messageEntry 4}

 messageCreationDate OBJECT-TYPE
    SYNTAX DateandTime
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
      "Time that the message was submitted
         to the first MTA"
::= {messageEntry 5}

messageSize OBJECT-TYPE
    SYNTAX Gauge32
    UNIT "Bytes"
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
      "Size in bytes of this Message"
 ::= {messageEntry 6}

 messagePriority OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
      "A string reprasentation for the priority
        of this message"
 ::= {messageEntry 7}

 messageMTASourceName OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
      "A string reprasentation for the name of the
        mta from wich the MS received the Message"
 ::= {messageEntry 8}

 messageMTASourceType OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
      "A string reprasentation for the type of the
        mta from wich the MS received the Message"
 ::= {messageEntry 9}

 messageMTASourceAgent OBJECT-TYPE
```

```
       SYNTAX DisplayString
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
         "A string reprasentation for the adress of the
          agent for the MTA appearing in the
          messageMTASource attribute"
   ::= {messageEntry 10}

   messageSupplementalInformation OBJECT-TYPE
       SYNTAX DisplayString
       MAX-ACCESS read-write
       STATUS current
       DESCRIPTION
         "A string reprasentation for a supplemental
          infomation to this Message"
   ::= {messageEntry 11}

   messageAccess OBJECT-TYPE
       SYNTAX Counter32
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
         "How many this message was accessed"
   ::= {messageEntry 12}

   messageAction OBJECT-TYPE
       SYNTAX Integer
       MAX-ACCESS read-write
       STATUS current
       DESCRIPTION
         "This is an action field. Possible values are:
           1 = delete the message
           2 = compress the message
           3 = encrypt the mesaage
           4 = send the message to the user specified
                 In the to field
           5 = send the message to the adress specified
                                            in the userHomeMailAdress Attribute
           6 = send the message to the adress specified
                  in the UserForeignMailAdress Attribute "
   ::= {messageEntry 13}

-- connectionTable
-- A table holding information about connections

connectionTable OBJCT-TYPE
        SYNTAX SEQUENCE OF ConnectionEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
             " The table holding information specific to a connection"
```

```
::= { ms 4 }

connectionEntry OBJCT-TYPE
        SYNTAX ConnectionEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
             " An entry associated with each connection "
::= { connectionTable 1}

ConnectionEntry ::= SEQUENCE {
                           connectionIndex Integer32,
                           connectionAverageSpeed Integer32,
                           connectionMechanism DisplayString,
                           connectionUpTime Integer32,
                           connectionClientHost DisplayString,
                           connectionInboundAbortReason DisplayString,
                           connectionInboundFailureReason DisplayString,
                           connectionRestartTime DateandTime,
                           connectionAbortTime DateandTime,
                           connectionAction Integer
}

    connectionIndex OBJECT-TYPE
       SYNTAX Integer32
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
         "An index to uniquely identify each
          connection for this ms"
    ::= {connectionEntry 1}

    connectionAverageSpeed OBJECT-TYPE
       SYNTAX Integer32
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
         " A string showing the connection
           speed between the client or
           mta and the ms measered in bytes/second"
     ::= {connectionEntry 2}

     connectionMechanism OBJECT-TYPE
       SYNTAX DisplayString
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
         " A string showing the connection
           mechanism beiing used"
     ::= {connectionEntry 3}
```

```
connectionUpTime OBJECT-TYPE
  SYNTAX Integer32
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
    "An integer containing the current
     connection time in seconds"
::= {connectionEntry 4}

connectionRemAppType OBJECT-TYPE
  SYNTAX DisplayString
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
    "A string showing the name of the
      client host"
::= {connectionEntry 5}

connectionInboundAbortReason OBJECT-TYPE
  SYNTAX DisplayString
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
    "The abort reason.
     An empty string indicates that
     the last attempt was successful. "
::= {connectionEntry 6}

connectionInboundRestartTime OBJECT-TYPE
  SYNTAX DateandTime
  MAX-ACCESS read-write
  STATUS current
  DESCRIPTION
    "The time when this connection
      was aborted.
      a zero value indicates
      the last attempt was successful. "
::= {connectionEntry 7}

connectionInboundAbortTime OBJECT-TYPE
  SYNTAX DateandTime
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
    "The time when this connection was aborted.
     a zero value indicates
    the last attempt was successful. "
::= {connectionEntry 8}

connectionInboundAbortReason OBJECT-TYPE
  SYNTAX DisplayString
```

```
      MAX-ACCESS read-only
      STATUS current
      DESCRIPTION
        "The failure reason.
         An empty string indicates that
         the last attempt was successful."
    ::= {connectionEntry 9}

    connectionAction OBJECT-TYPE
      SYNTAX Integer
      MAX-ACCESS read-only
      STATUS current
      DESCRIPTION
        "This is an action field. Possible values are:
          1 = abort this connection
          2 = restart this connection"
    ::= {connectionEntry 10}


-- userTable
-- A table holding information about users

userTable OBJCT-TYPE
        SYNTAX SEQUENCE OF UserEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
             " The table holding information specific to a user"
::= { ms 5 }

userEntry OBJCT-TYPE
        SYNTAX UserEntry
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
             " An entry associated with each user "
::= { userTable 1}

UserEntry ::= SEQUENCE {
                           userIndex Integer32,
                           userFolderList DisplayString,
                           userAction Integer,
                           userHomeMailAdress DisplayString,
                           userForeignMailAdress DisplayString
}


    userIndex OBJECT-TYPE
       SYNTAX Integer32
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
```

```
                  "The index for the user which is
                    equal to the userID"
    ::= {userEntry 1}

    userFolderList OBJECT-TYPE
       SYNTAX DisplayString
       MAX-ACCESS read-write
       STATUS current
       DESCRIPTION
                  "A sequence of strings with the identifiers of
                    all folders for this user include group folder"
    ::= {userEntry 2}

    userAction OBJECT-TYPE
       SYNTAX Integer
       MAX-ACCESS read-write
       STATUS current
       DESCRIPTION
                  "This is the action field. Possible values are
                    1 = add a user
                    2 = delete a user
                    3 = subscribe to a group folder
                    4 = unsubscribe from a group folder
                    5 = read-write in a group folder"
    ::= {userEntry 3}

    userHomeMailAdress OBJECT-TYPE
       SYNTAX IpAdress
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
                  "The mail adress of this user"
    ::= {userEntry 4}

    userForeignMail9Adress OBJECT-TYPE
       SYNTAX DisplayString
       MAX-ACCESS read-write
       STATUS current
       DESCRIPTION
                  "The foreign mail adress of this user
                    (e.g username@care-of-adress)"
    ::= {userEntry 5}

END
```

Literaturverzeichnis

[AB93] A. Achaya, B. R. Badrinath. *Delivering Multicast Messages in Networks with Mobile Hosts*. Dept. of Computer Science, Rutgers University, 13. Intl. Conf. on Distributed Computing Systems, May 1993.

[AB94] A. Achaya, B. R. Badrinath. *Checkpointing Distributed Application on Mobile Computer*. Dept. of Computer Science, Rutgers University, 3. Intl. Conf. on Distributed and parallel Information Systems, September 1994.

[AD93] A. Aziz, W. Diffie.*Privacy and Authentification for Wireless Local Area Networks*. Sunsofts, Inc. , Juli 1993.

[BB94] A. Bakre, B. R. Badrinath. *I-TCP : Indirect TCP for Mobile Hosts*. Dept. of Computer Science, Rutgers University, Piscataway, NJ 08855, DCS-TR-314, Oktober 1994.

[BBK94] B. Bellmann, S. Kümmel, T. Reigber, A. Schill (TU-Dresden). L. Heuser (Digital Equipement GmbH). R. kroh, D. Grill (Forschungszentrum Daimler Benz AG). *Systemunterstützung für verteilte Mobilrechner-Anwendungen*. 1994.

[BCF95] A. T. Boulontas, S. B. Calo, A. Finkel, I. Katzila. *Distributed Fault Identification in Telecommunication Networks*. Journal of Network and Systems Management, September 1995.

[DBP94] N. Davies, G. S. Blair, S. Pink. *Services to support distributed applications in a mobile environment*. SDNE'94, Juni 1994.

[DN94] A. DeSimone, S. Nanda.*Wireless Data : Systems, Standards, Applications*. AT&T Bell Laboratories, Holmdel, NJ0733-3030, Dezember 1994.

[Elk95] **draft-elkins-pem-pgp-02.txt**. *MIME Security with Pretty Good Privacy (PGP)*. M. Elkins; November 1995.

[EMA95a] Electronic Messaging Association (EMA). *Messaging Management Inplementator's Guide: Monotoring and Message Tracking*. TSC-Document, May 1995. `http://www.ema.org/html/pubs/tscmenu.htm`

[EMA95b] Electronic Messaging Association (EMA). *EMA's Messaging Management: User Requirements Document*. TSC-Document, May 1995. `http://www.ema.org/html/pubs/tscmenu.htm`

[FR94] W. Fumy, H. P. Rieß. *Kryptographie, Entwurf, Einsatz und Analyse*. R. Oldenburg Verlag, München, Wien, 1994.

[Gra95] Terry. Gray. *Message Access Paradigms and Protocols*. Technical Report, University of Wachington, September 1995.

[HA93] H.-G. Hegering, S. Abeck. *Integriertes Netz- und Systemmanagement*. Addison-Wesley, 1993.

[Hel95] A. Held. *Mobile Computing, Systeme, Kommunikation, Anwendungen*. International Thomson Publishing, 1995.

[HK95] S. Heilbronner, A. Keller. Weiterentwicklung der Agenten zum Management von Endsystemen und systemnahen Anwendungen. Bericht, Universität München, Dezember 1995.

[HN95] S. Heilbronner, B. Neumair. *Management mobiler Endsysteme: Anforderungen bei der Integration in bestehende Managementsysteme*. Bericht, Universität München, 1995.

[HNW95] H.-g. Hegering, B. Neumair, R. Wies. *Integriertes Management verteilter Systeme -Ein Überblick über State of the Art-* . LMU München, Bericht 9503, Januar 1995.

[HM96] **draft-ietf-asid-mime-person-00.txt**. *An Application/Directory MIME Content-Type White Pages Person Profile*. T. Howes, M. Smith; voraussichtlich Juli 1996.

[HR92] U. Hübner, F. Richter. *Charakteristika neuer E-Mail-Dienste und Architekturen* . TU Chemnitz, November 1992.

[IB95] T. Imielinski, B. R. Badrinath. *Mobile Wireless Computing : Challanges in Data Management*. Dept. of Computer Science, Rutgers University, New Brunswick, NJ 08903, Juni 1995.

[IDM91] D. Duchamp, I. Ionnidis, G. Marguire. *IP-Based Protocols for mobile Internetworking* . ACM SIGCOM Symposium on Communication, Architectures and Protocols , September 1991.

[IVB94] T. Imielinski, S. Vishwanatan, B. R. Badrinath. *Power efficient filtering of data on air*. EDBT, Cambridge, März 1994.

[JLLM94] R. Jain, Yi-Bing, Charles Lo, Seshadri Mohan. *A Caching Strategy to Reduce Network Impacts of PCS*. Bell Communication Research, Februar 1994.

[Jor95] E. Jordan.*Integrating E-Mail with X.500 Directories*. White Paper, Control Data Systems, Inc. 1995.

[Ker92] H. Kerner. *Rechnernetze nach ISO*. Addison-Wesley, 1994.

[KPS95] C. Kaufman, R. Perlman, M. Speciner. *Network Security Private Communication in a public Word*. Prentice Hall PTR, 1995.

[LMM95] Y. Lashkai, M. Metral, P. Maes. *Collaborative Interface Agents*. MIT Media Laboratory, Cambridge, MA 02139, 1995.

[Mey95] **draft-myers-imap-quota-00.txt**. *IMAP QUOTA Extention*. J. Meyers; Dezember 1995.

[Mis95] R. Miserre. *Datenfunk: Technik, Trends, Projekte*. Verlag Heiz Heise GmbH , 1995.

[MP91] M. Mouly, M. B. Paulet. *GSM Protocol Architecture: Radio Subsystem Signalling*. IEEE 41st Vehicular Technology Conference 1991.

[Per96] **draft-ietf-mobileip-protocol-16.txt**. *IP Mobility Support*. C. Perkins; April 1996

[Ros94] M. Rose. *An Introduction to Management of TCP/IP-based Internets*. . Prentice Hall, 1994.

[SKSW96] **draft-ietf-applmib-sysapplmib-01.txt**. *Definitions of Managed Objects for Applications*. J. Saperia, C. Krupczak, R. Sturm, J. Weinstock. Februar 1996.

[SNMP] SNMP ist in zahlreichen RFCs, diese sind : RFC1155, RFC1157, RFC1213, RFC1441-1452, RFC1901-RFC1908.
`http://www.nm.informatik.uni-muenchen.de/Forschung/snmp.html`
`http://wagner.nz.fh-koeln.de/rfc.html`

[Sto95] F. Stoll. *The Need for Decentralization and Privacy in Mobile Communication Networks*. Network Security Observations, Vol.1, No.1, Januar 1995.

[Web94] K. Weber. *von Electronic Mail zu multimedialer Post*. Informatik Spektrum, Band17, Heft4, August 1994.

[Wei95] J. Weinmiller. *A brief Introduction to the 802.11 draft standard for wireless LANs*. Internel Report, TU-Berlin, November 1995.

[X.400] **X.400** *Message Handling System: system and Service Overview*; CCITT Recommandations of the X.400 Series; 1992.

[X.500] **X.500** CCITT Recommandations of the X.500 Series; 1988.

[Zeg93] E. Zegwaart. *Privacy enhanced Mail in more Detail*. Computer Networks for Research in Europe, 1993.

[RFC821] **RFC 821** *Simple Mail Transfer Protocol (SMTP)*; J.B. Postel; August 1982.

[RFC822] **RFC 822** *Standard for the Format of ARPA Internet Text Messages*; D.H. Crocker; 1982.

[RFC1157] **RFC1157**. *Simple Network Management MIB (SNMP)*. J. Case, M. Fedor, M. Schoffstall, J. Davin. Mai 1991.

[RFC1213] **RFC1213**. *Management Information Base for Network Management of TCP/IP-based internets: MIB-II*. M. Rose, K. McCloghrie. März 1991.

[RFC1425] **RFC 1425** *SMTP Service Extentions*; J. Klensin, N. Freed, M. Rose, E.Stefferund, D. Crocker; Februar 1993.

[RFC1426] **RFC 1426** *SMTP Service Extention for 8bit-MIMEtransport* ; J. Klensin, N. Freed, M. Rose, E.Stefferund, D. Crocker; Februar 1993.

[RFC1460] **RFC 1460** *Post Office Protocol V3 (POP-3)*; M. Rose; Juni 1993.

[RFC1510] **RFC 1510** *KERBEROS (V5)*; J. Kohl, C. Neumann; September 1993.

[RFC1514] **RFC 1514** *Host Resources MIB*; P. Grillo, S. Waldbusser; September 1993.

[RFC1521] **RFC 1521** *Multipurpose Internet Mail Extentions (MIME)*; N. Borenstein, N.Freed; September 1993.

[RFC1565] **RFC 1565** *Network Services MIB*; S. Kille, N. Freed; Januar 1994.

[RFC1566] **RFC 1566** *Mail Monitoring MIB*; S. Kille, N. Freed; Januar 1994.

[RFC1567] **RFC 1567** *X.500 Directory Monitoring MIB*; S. Kill, G. Mausfie; Januar 1994.

[RFC1568] **RFC 1568** *Simple Network Paging Protocol V1 (SNPP V1)*; A. Gwinn Januar 1994.

[RFC1592] **RFC1592**. *Simple Network Management Protocol Distributed Protocol Interface Version 2.0*. B. Wijnen, G. Carpenter, K. Curran, A. Sehgal, G. Waters. März 1994.

[RFC1645] **RFC 1645** *SNPP V2*; A. Gwinn; July 1994.

[RFC1730] **RFC 1730** *Internet Message Access Protocol V4 (IMAP-4)*; M. Crispin; Dezember 1994.

[RFC1731] **RFC 1730** *IMAP4 Authentication Mechanisms*; J. Myers; Dezember 1994.

[RFC1732] **RFC 1730** *IMAP4 Compatibility with IMAP2 and IMAP2bis*; M. Crispin; Dezember 1994.

[RFC1733] **RFC 1730** *Distributed electronic mail models in IMAP-4*; M. Crispin; Dezember 1994.

[RFC1734] **RFC 1734** *POP3 Authentication command*; J. Meyers, C. Mellon; Dezember 1994.

[RFC1861] **RFC 1861** *SNPP V3*; A. Gwinn; Oktober 1995.

[RFC1869] **RFC 1869** *SMTP Service Extention for Message Size Declaration*; J. Klensin, N. Freed, M. Rose, E.Stefferund, D. Crocker; November 1995.

[RFC1894] **RFC 1894** *An Extensible Message Format for Delivery Status Notification* ; M.Moore,G.Vaudreuil. Januar 1996.