



Universität Karlsruhe (TH)  
Forschungsuniversität · gegründet 1825



Fakultät für **Informatik**

Institut für Telematik  
Cooperation & Management



# Evaluierung eines Nachfolgesystems für das Netzwerk- und System-Monitoring in der ATIS

Studienarbeit  
von

**Ingo Pansa**

Verantwortlicher Betreuer:  
Betreuender Mitarbeiter:

Prof. Dr. Sebastian Abeck  
Dipl.-Math. Klaus Scheibenberger

Bearbeitungszeit: 01. Februar 2007 – 11. Mai 2007



## Ehrenwörtliche Erklärung

Ich erkläre hiermit, die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet zu haben.

Karlsruhe, den 11. Mai 2007

---

Ingo Pansa



# Inhaltsverzeichnis

1	Einführung .....	7
1.1	Thematische Einordnung .....	7
1.2	Szenario: Fakultätsweiter Mailedienst.....	9
1.3	Ziel dieser Studienarbeit .....	11
1.4	Aufbau und Organisation dieser Arbeit .....	11
2	Modellierung eines realen Dienstes und Anforderungen an ein NMS .....	13
2.1	Systematisches Modell des Mailedienstes.....	13
2.2	Grundlegende Anforderungen an ein NMS .....	14
2.3	Management Architektur .....	14
2.3.1	Information Model .....	14
2.3.2	Organization Model .....	15
2.3.3	Communication Model .....	16
2.3.4	Function Model.....	17
2.4	Weitere Kriterien .....	20
2.5	Zusammenfassung .....	20
3	Ist-Szenario: BigBrother .....	21
3.1	Einführung .....	21
3.2	Aufbau und Funktionsweise .....	21
3.3	Die Agenten .....	23
3.4	Der Manager .....	23
3.5	Beispiel Mailsystem.....	24
3.6	Speicherung, Darstellung der gesammelten Information .....	26
3.7	Zusammenfassung BigBrother .....	28
3.8	Alternativen zu BigBrother.....	29
3.8.1	IBM Tivoli, openNMS.....	29
3.8.2	Nagios .....	29
3.9	Auswahl von Nagios für die Evaluation .....	30
4	Evaluation Nagios (Version 3) .....	31
4.1	Instanz einer Monitoring Architektur .....	31
4.2	Aufbau und Architektur von Nagios.....	32
4.2.1	Einleitung.....	32
4.2.2	Umsetzung des Monitoringsystems .....	32
4.2.3	Architektur von nagios.....	33
4.3	Umsetzung Mailsystem.....	35
4.3.1	Host - Infrastrukturkomponente – Switch .....	35
4.3.2	Host - Infrastrukturkomponente – Firewall .....	35
4.3.3	Host - Mailtransportserver – iramx1 .....	36
4.3.4	Hostgruppen.....	36
4.3.5	Servicegruppen .....	36
4.4	Datenspeicherung und Visualisierung .....	36
4.4.1	Datenspeicherung.....	36
4.4.2	Visualisierung .....	37
4.5	Webinterface .....	38
4.6	Verwaltung von nagios .....	41
4.6.1	Installation und Konfiguration.....	42
4.6.2	Konfigurationsinterface fruity (kooperierender Ansatz) .....	42
4.6.3	Groundworks Opensource Monitor (integrierter Ansatz).....	43
4.7	Weiterführende Möglichkeiten .....	44
4.8	Zusammenfassung .....	44

4.9	Konkrete Umsetzung der Monitoring-Architektur.....	45
5	nagios und Dienstorientierung .....	47
5.1	Dienstorientierung – der Dienst im Mittelpunkt .....	47
5.2	Eigenschaften eines IT-Dienstes .....	49
5.3	Anforderungen zur Umsetzung eines Dienstmodells.....	50
6	Zusammenfassung, Ausblick .....	53
6.1	Zusammenfassung.....	53
6.2	Ausblick .....	53
7	Anhänge .....	56
7.1	Abbildungsverzeichnisse (Abbildungen und Tabellen) .....	56
7.2	Index.....	57
7.3	Abkürzungen und Glossar .....	59
7.4	Literatur und Quellenverweise .....	63

# 1 Einführung

## 1.1 Thematische Einordnung

Sowohl Firmen wie auch Einrichtungen im nicht-kommerziellen Bereich werden in ihren Geschäftsprozessen immer mehr durch IT-Systeme unterstützt. Steigende Anforderungen an die IT erhöhen aber auch gleichzeitig die Komplexität der verwendeten Systeme. Um dieser Komplexität zu begegnen, wird der Dienst als Zusammenfassung der funktionalen und qualitativen Anforderung in den Vordergrund gestellt, und damit technische, für einen Kunden schwer verständliche Parameter, dahinter verborgen. Zwischen einem *Dienstnehmer* auf der einen Seite und einem *Dienstanbieter* auf der anderen Seite bildet sich eine *Dienstleistungsbeziehung*, welche mittels *Dienstleistungsvereinbarungen* (Service Level Agreements, SLA's) abgesichert und garantiert werden. Als Konsequenz daraus entsteht der Bedarf, die Dienstleistung nicht nur *quantitativ* (funktional), sondern auch *qualitativ* bewerten zu können. Dazu müssen jedoch die Eigenschaften eines Dienstes gemessen und überwacht werden, die gegenüber einem Kunden zu gewährleisten sind.

Die vorliegende Arbeit betrachtet ausschließlich (software-)Systembasierte Dienste (so genannte IT-Dienste). Die Erweiterung eines Systembasierten Dienstes um Zusatzdienste wie beispielsweise Schulungsdienste oder Supportdienste wird in dieser Arbeit nicht thematisiert.

IT-Dienste werden durch Aggregation von vorhandenen IT-Ressourcen wie Netzwerke, Anwendungen oder Systeme erreicht, oder anders ausgedrückt: Die Dienste werden von Ressourcen der IT-Infrastruktur erbracht. Als Grundvoraussetzung für eine erfolgreiche Dienstleistung muss das IT-Management daher eine funktionierende Infrastruktur sicherstellen können.

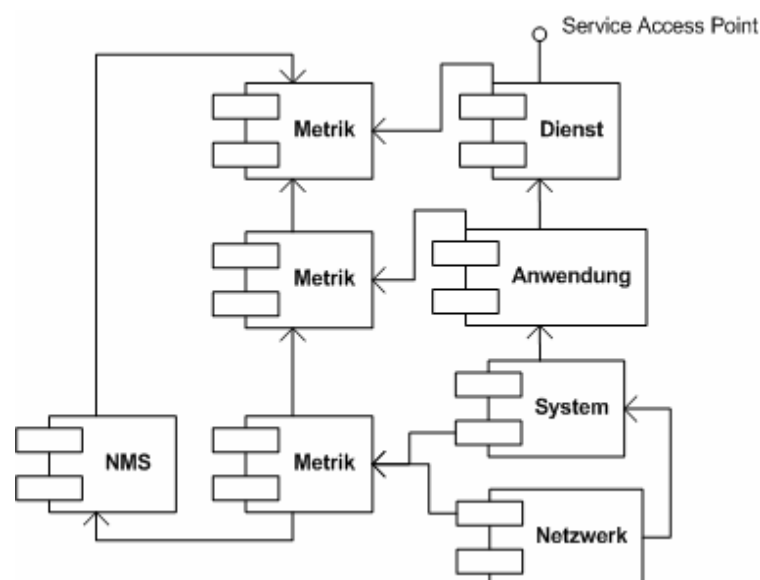


Abbildung 1 prinzipieller Aufbau eines Dienstes

Unter dem Begriff „IT-Management“ versteht man alle Maßnahmen, die für einen zielorientierten, effektiven und effizienten Betrieb eines verteilten Systems und seiner

Ressourcen benötigt werden[1].

Entsprechend ihrer hierarchischen Anordnung lassen sich folgende Disziplinen im IT-Management unterscheiden:

- *Dienstmanagement* betrachtet die funktionale und qualitative Leistungserbringung eines systembasierten Dienstes und evtl. von Zusatzdiensten. Das Dienstmanagement umfasst jene Verfahren, welche zur Vereinbarung und zur Überwachung der Einhaltung der Dienstleistungsverträge benötigt werden.
- *Anwendungsmanagement* betrachtet einzelne Anwendungen als logische Ressourcen und die damit verknüpften Verfahren und Maßnahmen. Ein Teil des Anwendungsmanagements ist unter anderem die Anwendungsüberwachung zur Laufzeit der Anwendung. Das Anwendungsmanagement geht darüber aber hinaus und betrachtet auch den kompletten Entwicklungs- und Lebenszyklus einer Anwendung.
- *Systemmanagement* bezieht sich auf Systeme als Ganzes, beispielsweise einen Router, oder einen Server, aber auch auf einen Systemdienst in Form eines Dämons wie LDAP. Im Rahmen des Systemmanagements erfolgt unter anderem die Erfassung von CPU-Auslastung, Speicherbelegung von Systemprozessen.
- *Netzwerkmanagement* bezieht sich auf Maßnahmen, um die Infrastruktur eines Netzwerkes in Betrieb zu halten. Im Rahmen des Netzwerkmanagements erfolgt beispielsweise die Messung und Steuerung der Verfügbarkeit oder des Durchsatzes einzelner Netzwerkkomponenten.

In jeder dieser Ebene lassen sich die darin anfallenden Managementaufgaben weiter strukturieren. Es werden üblicherweise fünf typische Aufgabenfelder identifiziert:

- Fehlermanagement
- Konfigurationsmanagement
- Abrechnungsmanagement
- Leistungsmanagement
- Sicherheitsmanagement

Wegen der Initialen der englischen Wörter dieser Begriffe sind diese Aufgaben unter dem Begriff FCAPS bekannt (*Fault-, Configuration-, Accounting-, Performance-, Securitymanagement*) [1].

Konzentriert man sich auf die Ebenen Netzwerk und System, dann unterstützt ein Netzwerk- und System-Managementsystem (NMS) die obigen Aufgabenfelder, indem messbare Parameter von Netzwerkkomponenten und Systemen (so genannte *Managed Nodes*) überwacht und damit Informationen über den Zustand dieser Komponenten geliefert werden (*Monitoring*). Aufgrund dieser Zustandsinformationen können dann (evtl. automatisierte) Aktivitäten im Prozess des Netzwerk- und System-Management angestoßen werden (*Controlling*), wenn beispielsweise definierte Schwellwerte überschritten werden.

Die Verfügbarkeit einer Ressource, beispielsweise eines Servers oder Systemdienstes, kann unabhängig vom Kontext des Dienstes, in den die Ressource eingebunden ist, durch ein NMS überwacht werden. Um einen (abstrakten) Dienst anbieten zu können,



stellt sich jedoch die Frage, inwiefern und auf welche Weise Informationen, die durch ein Netzwerk- und System-Managementsystem (NMS) erfasst werden, mit zur Beurteilung der Qualität eines Dienstes herangezogen werden können.

## 1.2 Szenario: Fakultätsweiter Mailedienst

Die Abteilung Technische Infrastruktur (im folgenden ATIS genannt) stellt für die Einrichtungen der Informatikfakultät an der Universität Karlsruhe und deren Mitarbeiter neben verschiedenen systembasierten Diensten wie Mail oder VPN auch den Zugang zum *Lokalen Informatik Netz Karlsruhe* (LINK) mit Anbindung an das weltweite Internet bereit. Als integraler Teil des universitätsweiten Netzes erstreckt sich das LINK über den ganzen Campus; einzelne Dienstnehmer sind an unterschiedlichen Positionen lokalisiert. Für die zuverlässige Erfüllung der Dienstangebote der ATIS ist eine funktionierende Netzinfrastruktur somit unerlässlich. Zum Verständnis über den physikalischen bzw. logischen Aufbau des Datennetzes sei hier auf [2] verwiesen.

Als reales Szenario für die Evaluierung eines Systems für das Netzwerk- und System-Management wird beispielhaft der Email-Dienst der ATIS betrachtet. Anhand dieses Beispiels werden im Laufe der Arbeit detaillierte Anforderungen an ein NMS abgeleitet. Anhand dieser Anforderungen wird dann ein ausgewähltes Produkt evaluiert. Daher wird im Folgenden dieses Szenario kurz vorgestellt.

Im komplexen Beispiel des Email-Dienstes müssen aus Sicht des IT-Managements vier Systeme mit unterschiedlichen Systemdiensten überwacht werden. Ein- bzw. ausgehender Mailverkehr wird über einen von zwei möglichen sog. Mail-Exchangern (mx1, mx2) geleitet. Auf diesen laufen als wichtigste Dienste zum einen jeweils eine Instanz eines sog. Mailtransfer-Agent (MTA), der für das Weiterleiten von Emails mittels des standardisierten SMTP [3] zuständig ist (im Falle der ATIS wird hier das Opensource Produkt *Exim* [4] eingesetzt), zum anderen eine lokale Replikation eines LDAP-Verzeichnisses. Letztere wird ständig mit Änderungen aus dem zentralen Verzeichnisdienst (LDAP) synchronisiert und dient der Überprüfung durch den MTA, ob für eine eingehende Mail der in der Zustellinformation der Mail adressierte Empfänger in der Fakultät überhaupt vorhanden ist. Wird eine Email für zustellbar befunden, wird aufgrund der Zustellinformation das System ermittelt, auf dem der Nutzer sein Mailverzeichnis hat. Auch diese Information wird aus dem lokalen LDAP-Verzeichnis bezogen, in der Regel ist dies der zentrale Mailserver (irams1) der ATIS. Befindet sich das Postfach des Empfängers aber nicht auf dem zentralen Mailserver wird die Email zu dem zuständigen Institutsmailserver weitergeleitet. Diese Institutsmailserver liegen nicht in der Administrationshoheit der ATIS.

Da die Belastung durch Viren oder Spam in Emails aus dem Internet immer stärker zunimmt, wird auf den beiden Mail-Exchangern jeglicher Mailverkehr noch durch einen Virens Scanner geführt, beziehungsweise durch Spamfilterung von der weiteren Zustellung ausgeschlossen.

Auf dem zentralen Mailserver (ms1) laufen eine oder mehrere Instanzen eines IMAP-Servers, im Falle der ATIS wird hier das Opensource-Produkt *Courier* eingesetzt. Dieser IMAP-Server ermöglichen es mittels des POP- oder IMAP- Protokolls [5][6] einem Nutzer, nach seiner Authentifizierung gegenüber dem LDAP-Verzeichnisdienst, das Lesen seiner empfangenen Emails. Der Nutzer kann hierfür verschiedene Email-Klientenprogramme verwenden, zusätzlich besteht noch die Möglichkeit, Emails über ein Webinterface [7] abzurufen bzw. zu erstellen und zu versenden.

Abbildung 2 zeigt vereinfacht den Aufbau des E-Mail-Systems.

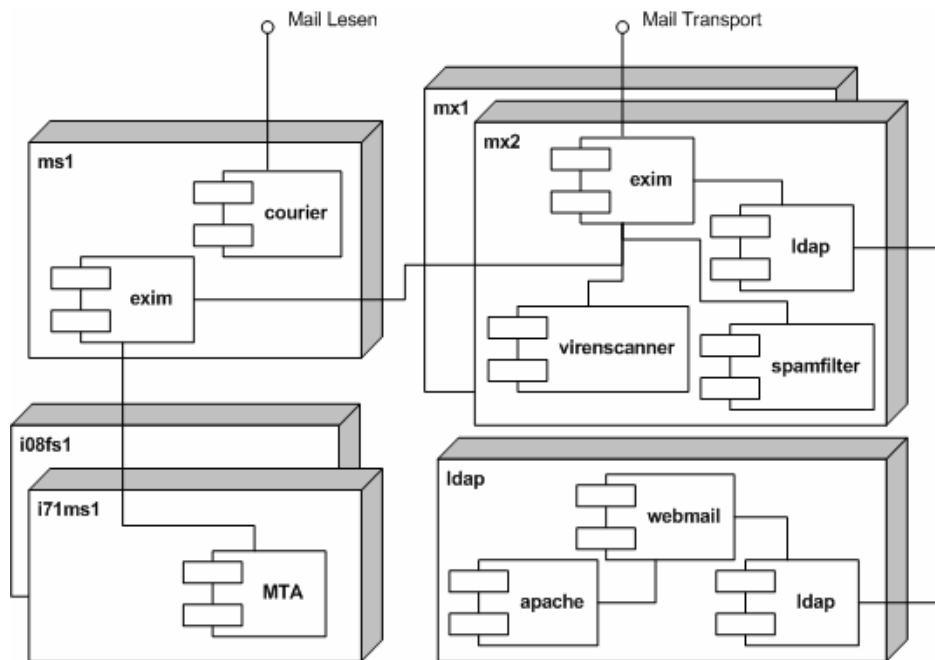


Abbildung 2 Vereinfachter Aufbau des Mailsdienstes

Wie bereits erwähnt, benötigt der Systemdienst Exim einen funktionierenden Verzeichnisdienst, um feststellen zu können, ob ein Empfänger in der ATIS vorhanden ist und auf welchem Mailserver dessen Mailverzeichnis lokalisiert ist.

Der Verzeichnisdienst basiert auf dem Opensource-Produkt *openldap* [8], dessen zentralen Datenbestand zur Steigerung der Ausfallsicherheit auf mehrere Replikate verteilt wird. Lesezugriffe werden dann gegen ein jeweils lokales Replikat durchgeführt, Schreibzugriffe werden zur Sicherung der Datenkonsistenz nur auf dem zentralen LDAP-Server ausgeführt und dann wiederum an die Replikate verteilt.

Der LDAP-Verzeichnisdienst zeichnet sich durch die Tatsache aus, dass ein Benutzer normalerweise nicht direkt mit diesem Dienst in Berührung kommt (im Vergleich zum Email-Dienst) – die in den Mailsdienst eingebundenen Systeme kommunizieren jedoch intensiv mit dem Verzeichnisdienst. Man kann in diesem Falle von einem so genannten *Basisdienst* sprechen, der auch für viele weitere Dienste innerhalb der ATIS benötigt wird.

Die am Ende von Abschnitt 1.1. gestellte Frage, inwiefern es sinnvoll bzw. notwendig ist, Messungen eines NMS unterstützend zur Beurteilung der Qualität eines Dienstes heranzuziehen, soll kurz an dem Beispiel des Mailsdienstes verdeutlicht werden:

- Fällt beispielsweise eine Netzkomponente (ein Switch oder Router) zwischen einem der beiden Mail-Exchangern und dem Mailserver aus, kann durch die entkoppelte Betrachtung der Systeme der Mailinfrastruktur nicht festgestellt werden, dass der Benutzer keine Emails mehr zu Zielen versenden kann, die außerhalb der Zuständigkeit der ATIS liegen, während der interne Mailverkehr dadurch evtl. nicht gestört wird. Vielmehr muss in die Betrachtung das Wissen über den Zusammenhang der Mailinfrastruktur, hier der beteiligten Netzwerkkomponenten, einbezogen werden, um eine Fehlfunktion des Dienstes einzugrenzen.
- Umgekehrt stellt sich die Frage welche Dienste durch den Ausfall einer

bestimmten Ressource der IT-Infrastruktur betroffen sind. Dies kann beispielsweise Einfluss auf die Abfolge von Aktivitäten zur Behebung der Störung haben. Das bedeutet, dass auch hier die Kenntnis der Beziehung einer Komponente zu einem Dienst wichtig ist.

- Email stellt einen Nachrichtenorientierten, unzuverlässigen Kommunikationsdienst dar. Der Nutzer dieses Dienstes hat jedoch Interesse an einer zeitnahen und zuverlässigen Zustellung seiner Mail. Ein für einen Nutzer interessanter Parameter ist daher beispielsweise die durchschnittliche Zustellzeit einer Mail (Zeit vom Beginn des Empfangs der Email bis zur Auslieferung in das Postfach des Nutzers). Da hier die Auslastung der System- und Netzwerkinfrastruktur wesentlichen Einfluss nimmt, wäre es hilfreich, Messwerte entsprechender Parameter in den Kontext des Dienstes Mail stellen zu können. Die Ermittlung einer solchen Qualitätsinformation würde aber bereits die Korrelation verschiedener Messwerte über Systemgrenzen hinweg erfordern.

### 1.3 Ziel dieser Studienarbeit

Während das unabhängige Überwachen einer Vielzahl von physikalischen oder logischen Parametern von Netzwerk- oder Systemkomponenten heute kein Problem darstellt, wird nun gefordert, Netz- und Systemüberwachung in Beziehung zu den von den Ressourcen unterstützten Diensten zu stellen. Dies, um einerseits Dienste garantiert anbieten zu können, diese aber auch andererseits einem Nutzer gegenüber belegen zu können.

Im Rahmen dieser Studienarbeit werden somit zwei Ziele verfolgt:

- Die bestehende Infrastruktur zur Überwachung der Systeme und Netzwerkkomponenten in der ATIS soll aktualisiert werden (Monitoring-Architektur). Dazu ist es erforderlich eine Evaluierung aktueller (Opensource) NMS-Produkte durchzuführen. Konkrete Anforderungen sind zu identifizieren und für ein in Frage kommendes System ist anhand einer Testinstallation die Einbindung in den Betrieb zu verifizieren.
- Es wird die Frage untersucht, inwieweit ein solches System eine Dienstorientierung der Managementinformationen ermöglicht, d.h. inwieweit lassen sich Messwerte aus dem System für das Netzwerk- und System-Management dem Kontext eines Dienstes sinnvoll zuordnen. Dafür wird das in Kapitel 1.2 eingeführten Szenario des Mailedienstes im Laufe dieser Arbeit herangezogen und analysiert.

### 1.4 Aufbau und Organisation dieser Arbeit

Kapitel 1 führt in die Thematik Network-and-System-Management Systeme (NMS) ein. Es werden grundsätzliche Begriffe definiert, anschließend wird der Mail-Dienst als zentraler ATIS-Dienst beschrieben. Abschnitt 3 formuliert die Ziele dieser Studienarbeit.

Zu Beginn von Kapitel 2 werden generelle Anforderungen an ein NMS beschrieben.

Kapitel 3 analysiert das bestehende System BigBrother und beschreibt dessen Aufbau und Funktionsweise. Mithilfe der im zweiten Kapitel aufgestellten Anforderungen wird die Notwendigkeit aufgezeigt, warum BigBrother durch ein moderneres NMS abgelöst werden soll, anschließend wird eine Kurzübersicht über verfügbare Produkte aus dem Bereich Netzwerk- und Systemmanagement gegeben.

Im 4. Kapitel wird nagios für einen Einsatz in der ATIS evaluiert werden. Als Grundlage dient das bereits in diesem Kapitel eingeführte Mailsystem, welches im Laufe dieser Arbeit immer weiter verfeinert wird.

Kapitel 5 führt in die Thematik der Dienstorientierung ein. Neben der Definition von Eigenschaften eines IT-Dienstes wird anhand des Mailsystems gezeigt, wie mit Hilfe eines NMS Messwerte der Ressourcenebene in den Kontext eines Dienstes gestellt werden können.

Das letzte Kapitel schließt mit einer Zusammenfassung der Ergebnisse dieser Arbeit ab und formuliert einige weitergehende Möglichkeiten (im Hinblick auf automatisiertes Management ).

**Tabelle 1 Aufbau und Organisation der Arbeit**

Aufbau und Organisation	
Kapitel 1	Thematische Einordnung Einführung Mailsystem Ziel der Arbeit Aufbau und Organisation
Kapitel 2	Modellierung des Mailsystems Grundlegende Anforderungen Management Architektur
Kapitel 3	Ist-Zustand BigBrother
Kapitel 4	Evaluation nagios (Version 3 ) Migrationsstrategie
Kapitel 5	nagios und Dienstorientierung
Kapitel 6	Zusammenfassung und Ausblick
Kapitel 7	Anhänge Abbildungsverzeichnis Index Abkürzungen und Glossar Literaturverzeichnis

## 2 Modellierung eines realen Dienstes und Anforderungen an ein NMS

Ziel dieses Kapitels ist es, grundlegende Anforderungen an ein NMS zu formulieren. Zunächst wird dazu das in Kapitel 1 beschriebene Mailsystem mit dessen Abhängigkeiten zwischen den verschiedenen Systemprozessen („Systemdiensten“) modelliert, um daraus Anforderungen an ein NMS abzuleiten. Mit diesen Anforderungen kann dann im folgenden Kapitel eine Bewertung des aktuellen System BigBrother und von Alternativsystemen erfolgen.

### 2.1 Systematisches Modell des Mailsdienstes

Die folgende Abbildung 3 stellt den Zusammenhang der unterschiedlichen am Mailsdienst beteiligten Systemprozesse dar. Dabei wurde bereits eine Aufteilung in verschiedene Teildienste *Mail-Transport*, *Mail-Lesen* und *Verzeichnisdienst* vorgenommen.

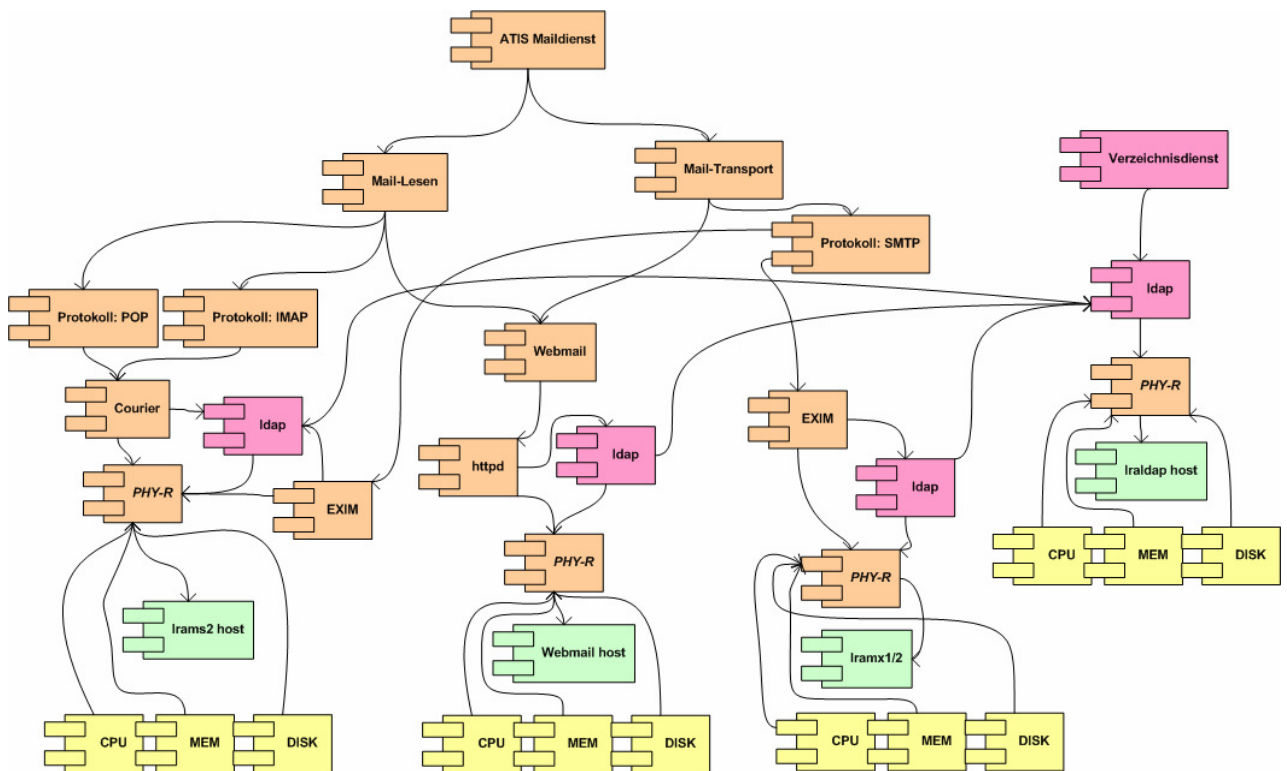


Abbildung 3 Systematischer Aufbau des Mail-Dienstes

Es ist ersichtlich, dass für die Erfassung dieses noch „überschaubaren“, aber dennoch schon recht komplexen verteilten Dienstes bereits eine Vielzahl unterschiedlicher Systeme und Systemdienste überwacht werden müssen. Die sich hierbei ergebenden Abhängigkeiten müssen miterfasst werden. Bezieht man noch die institutseigenen Mailserver mit ein, müssen auch unterschiedliche Administrationsdomänen, also entsprechende Organisationsstrukturen, im Rahmen dieses Dienstes berücksichtigt werden. Die zentrale Frage wird daher sein, welche Methoden ein NMS bietet, um diese komplizierte Informationsmenge zu strukturieren.

## 2.2 Grundlegende Anforderungen an ein NMS

Ein NMS verwaltet die Ressourcen wie Netzwerkgeräte, Server oder Systemprozesse einer Infrastruktur. Dabei ist es unerheblich, ob diese Ressourcen realer (CPU, Arbeitsspeicher, ...) oder logischer Natur (Systemprozesse) sind (s. Abbildung 3). Der Begriff der *Ressource* steht daher im Mittelpunkt eines NMS. Kernbestandteil dieser Verwaltung ist zunächst die *Überwachung* (Monitoring) der Ressourcen.

## 2.3 Management Architektur

Zunächst wird eine generelle Überwachungs- und Steuerungsarchitektur (*Management-Architektur*) vorgestellt. Ein *reines* Überwachungssystem, welches keinerlei Steuerungsmöglichkeiten bietet (reines Monitoring), ist als Teilmenge dieser generellen Architektur aufzufassen. Im Rahmen dieser Arbeit wird, wie oben bereits angedeutet, ausschließlich der Monitoringaspekt betrachtet. Es wird davon ausgegangen, dass der steuernde Eingriff durch einen Administrator unmittelbar auf den Netzwerkkomponenten und Systemen erfolgt. Als Handlungsgrundlage dient dazu die vom NMS bereitgestellte Information.

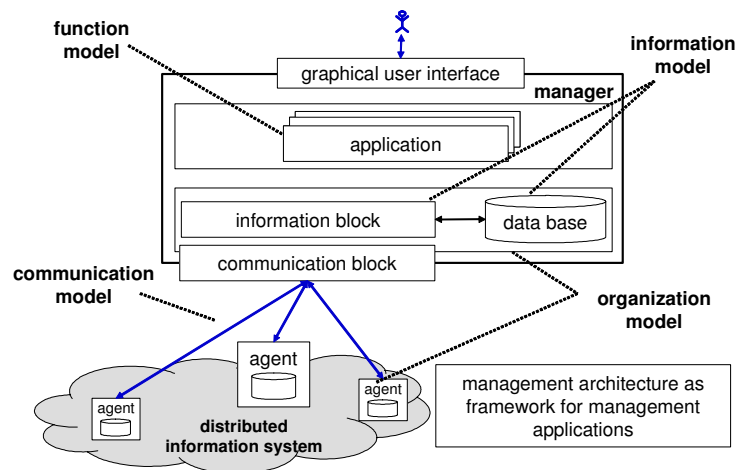


Abbildung 4 Generelles Modell zur Beschreibung einer Überwachungsarchitektur

Das in **Abbildung 4** vorgestellte generelle Modell [9] besteht aus den Komponenten *Manager*, *Agent*, *Database* und *Userinterface*. Modellbeziehungen (*Function Model*, *Information Model*, *Communication Model* und *Organization Model*) gewährleisten die Interoperabilität verschiedener Systeme, ermöglichen den direkten Systemvergleich und erleichtern die Entwicklung von offenen Protokollen zur Kommunikation.

Ausgehend von diesen vier Modellbeziehungen werden nun allgemeine Anforderungen an ein NMS formuliert.

### 2.3.1 Information Model

#### 2.3.1.1 Datenspeicherung

Zur erfolgreichen Weiterverarbeitung der gesammelten Information müssen die Mess- und Ereignisdaten entsprechend abgelegt werden. Es ist dabei von Vorteil, wenn auch hier standardisierte Verfahren zum Einsatz kommen, beispielsweise Data-Warehouse Systeme, CIM [11] oder ähnliches.

**Forderung 1:**

Die gesammelten Messdaten müssen in einer geeigneten Weise zur Weiterverarbeitung (Abrechnung, ...) abgelegt werden. Standardisierte Verfahren sind hierbei zu bevorzugen.

**2.3.1.2 Fehlerzustände**

Um Fehler unabhängig von einer fehlerhaften Ressource beschreiben zu können, ist es vorteilhaft, so genannte Fehlerzustände einzuführen.

Diese Fehlerzustände sind für alle Ressourcen verbindlich und erweitern das Konzept der Modularität. Dabei wird nicht die Art oder Qualität eines Fehlers angegeben, vielmehr werden im Zusammenhang mit Schwellwertangaben verschiedene Fehlergrade definiert.

Zumindest sollten die Zustände *erreichbar* und *nicht-erreichbar* identifizierbar sein. „Erreichbarkeit“ bezieht sich hierbei sowohl auf die physikalische Erreichbarkeit eines Hosts (mittels Netzwerkping) als auch auf die Erreichbarkeit eines Systemprozesses (ping an Port 80 prüft die Erreichbarkeit des http-Systemprozesses). Um eine feinere Abstufung zu ermöglichen, können Zwischenzustände beschrieben werden. *Warnung* bzw. *Kritisch* könnten bei drohender Überlast einen zuständigen Administrator rechtzeitig vor einem aufkommenden Problem alarmieren.

**Forderung 2:**

Fehler sollen in Form von Fehlerzuständen gemeldet werden. Dadurch erreicht man auch eine ressourcenunabhängige Möglichkeit, Fehler zu beschreiben.

**2.3.2 Organization Model****2.3.2.1 Manager und Agenten**

Bei der Beschreibung des Aufbaus eines NMS haben sich die Begriffe *Manager* und *Agent* durchgesetzt. Unter einem Agent versteht man eine Softwarekomponente, welche stellvertretend für einen Manager auf einem *Managed Node* agiert. Die zu überwachenden Eigenschaften werden hinter *Managed Objects* verborgen und sind einem Managed Node zugeordnet.

Der Manager wiederum stellt die gesammelte Information einem Benutzer zur Verfügung, oder stößt unter Umständen bestimmte Aktionen an, um einen definierten Zustand wiederherzustellen.

**Forderung 3:**

Das NMS muss klar strukturiert aufgebaut sein. Eine Trennung von Managern und Agenten und eine Kapselung von zu überwachenden Eigenschaften in Managed Objects werden daher gefordert.

**2.3.2.2 Strukturierung**

Um eine Infrastruktur effizient zu überwachen, ist es vorteilhaft, deren Aufbau so natürlich wie möglich auf das NMS zu übertragen. Idealerweise können Systemprozesse einem Host zugeordnet werden (*implizite* Gruppierung), aber auch beliebige Systemprozesse unterschiedlicher Hosts sollten zu einer Gruppe zusammengeschlossen werden können (*explizite* Gruppierungen). Dadurch entsteht die

Möglichkeit, Ressourcen (logische wie physikalische) zu gruppieren und aggregiert in die Überwachung aufzunehmen.

Ebenso muss die gesammelte Information entsprechend des modellierten Netzwerks strukturiert für externe Weiterverarbeitung abgelegt werden.

*Forderung 4:*

Die zu überwachende Infrastruktur muss so natürlich wie möglich modelliert werden können. Dazu ist es notwendig, logische Gruppierungen der Infrastruktur auf das NMS übertragen zu können.

### 2.3.2.3 Abhängigkeiten

Abhängigkeiten in einer Infrastruktur sind offensichtlich (siehe auch im Modell des Mailsdienstes), daher muss ein NMS diese Abhängigkeiten auch erfassen können. Man unterscheidet zwischen *impliziten* und *expliziten* Abhängigkeiten.

Erstere treten immer dann auf, wenn Infrastrukturkomponenten physikalisch direkt voneinander abhängen. Als Beispiel kann hier ein Switch betrachtet werden. Der Fehlerfall „Switch fällt aus“ ist implizit mit dem Fehlerfall „Server nicht erreichbar“ verbunden.

Explizite Abhängigkeiten werden vom Administrator formuliert und beziehen sich auf logische Abhängigkeiten in der Infrastruktur. Hierunter fallen jegliche Ressourcen eines Netzwerkes, also Hostsysteme gleichermaßen wie auch Systemprozesse auf unterschiedlichen Hosts.

Beispiele hierfür können im Mailsystem der ATIS gefunden werden. Bei der Zustellung einer E-Mail in das Postfach eines Empfängers muss dessen zuständiger Mailserver ermittelt werden. Diese Information ist im zentralen Verzeichnisdienst LDAP abgelegt. Somit besteht also eine explizite Abhängigkeit zwischen der korrekten Zustellung einer E-Mail durch den Systemprozess *exim* und der Verfügbarkeit des Systemprozesses *ldap*.

*Forderung 5:*

Die in der Infrastruktur auftretenden Abhängigkeiten müssen durch das NMS abgebildet werden können. Implizite Abhängigkeiten sollten automatisch gebildet werden, explizite Abhängigkeiten sollten frei zwischen beliebigen Ressourcen formulierbar sein.

## 2.3.3 Communication Model

### 2.3.3.1 Zugriff auf Information

Um eine Kommunikation innerhalb der Managementarchitektur zwischen Manager und Agent zu gewährleisten, müssen offene und transparente Protokolle zum Einsatz kommen. Das Protokoll muss ohne jegliches Wissen über Implementierungsdetails des Managers oder der Agenten arbeiten. Mit dem *Simple Network Management Protocol* (SNMP [10]) existiert ein industrie-offener Standard, der die Kommunikation zwischen Manager und Agenten regelt.

*Forderung 6:*

Die Kommunikation zwischen Managern und Agenten muss durch offene Protokolle möglich sein. Nur dadurch kann eine maximale Interoperabilität zu anderen Systemen gewährleistet werden.

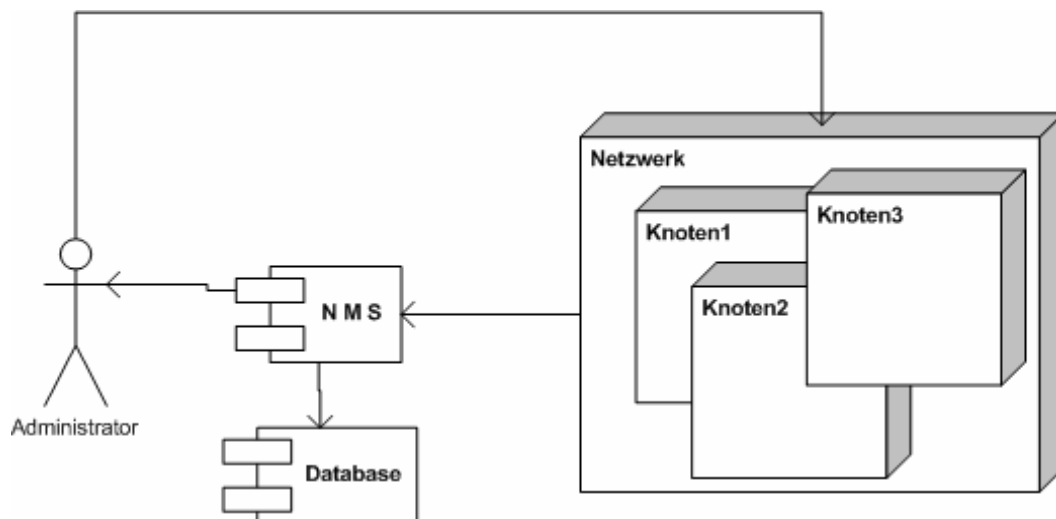


## 2.3.4 Function Model

### 2.3.4.1 Einordnung in FCAPS

Ein NMS hat zur Hauptaufgabe, einen reibungslosen Betrieb der Netzinfrastruktur zu gewährleisten. Wie bereits in Kapitel 1 dargestellt, lassen sich auch auf der Ebene der Netz- und Systeminfrastruktur die Managementaufgaben in die folgenden fünf Kategorien (FCAPS) differenzieren. In Bezug auf den Fokus der Arbeit können hier einige Einschränkungen vorgenommen werden.

*Fehlermanagement* - Das Erkennen und Melden von Fehlern auf Ressourcenebene gehört zu den Hauptaufgaben eines NMS. Während Fehler zu beliebigen Zeitpunkten auftreten können, wirken diese sich jedoch erst während der Nutzung einer Infrastrukturkomponente aus. Es müssen also insbesondere Funktionen zur Verfügung gestellt werden, um Fehler zu erkennen, darauf zu reagieren und einen definierbaren Zustand wiederherstellen zu können. Weiterhin sollte es möglich sein, die Infrastruktur (Hosts) und die drauf laufenden Systemdienste (Services) getrennt voneinander betrachten zu können. Dabei wird vorausgesetzt, dass bei Eintritt eines Fehlers der Administrator direkt auf die fehlerhaften Ressourcen einwirkt, um den Fehlerzustand zu beheben. **Abbildung 5** verdeutlicht diesen so genannten open-control loop.



**Abbildung 5** open-control loop

*Leistungsmanagement* ist als konsequente Ergänzung des Fehlermanagements anzusehen. Da das Leistungsmanagement qualitative Aspekte in den Vordergrund stellt, ist es wichtiger Bestandteil zur Unterstützung von Dienstgüte. Während das Fehlermanagement dafür verantwortlich ist, dass die Infrastruktur überhaupt verfügbar ist, geht das Leistungsmanagement einen Schritt weiter und versucht qualitative Parameter der Infrastrukturrressourcen zu erfassen und auf diese einzuwirken [1].

Anforderungen an das Leistungsmanagement sind daher weniger in der Reaktion auf Fehlerzustände zu suchen, als vielmehr in der Reaktion auf sich verändernde Leistungsdaten von Komponenten (z.B. Antwortzeiten). Um flexiblen Qualitätsvorgaben von der Dienstschnittstelle eines angebotenen Dienstes begegnen zu können, müssen also auch Mechanismen im NMS zur Verfügung stehen, um mittels den

gesammelten Messdaten die Netz- und Systemkomponenten dahingehend zu überwachen, dass die Vorgaben der Dienstschnittstelle erreicht werden.

Wie diese Vorgaben genau definiert werden, ist nicht Bestandteil des Leistungsmanagements zur Laufzeit.

*Konfigurationsmanagement* umfasst auf Netz- und Systemebene die Anpassung der Ressourcenkonfiguration (physikalische wie logische) an die aktuellen Betriebsbedingungen (Routingtabellen, Warteschlangenlänge, ...).

Es ist also insbesondere mit dem Fehlermanagement eng verbunden, da zur Laufzeit im Fehlerfall (oder drohenden Fehlerfall) häufig aktuelle Konfigurationsparameter benötigt werden, um korrekte Fehlerbehebungsmaßnahmen einzuleiten

*Abrechnungsmanagement* befasst sich mit der Abrechnung angefallener Dienstleistungskosten, bezogen auf Netz- und Systemebene also auf angefallene Rechenzeit, verbrauchter Speicher, etc. Diese Aufgabe ist jedoch aus Sicht eines NMS als eher zweitrangig anzusehen, im Rahmen des Abrechnungsmanagements muss ein NMS lediglich geeignete Messdaten zur Verfügung stellen, mit deren Hilfe dann für benutzte Dienstleistungen abgerechnet werden kann.

*Sicherheitsmanagement* auf Netz- und Systemebene beschränkt sich wie das Abrechnungsmanagement auch auf die Sammlung von sicherheitsrelevanten Daten (Login-Protokolle, fehlgeschlagene Fernfunktionsaufrufe, etc.) und wird daher nicht weiter verfolgt.

Somit folgt also:

*Forderung 7:*

Die Hauptaufgabe eines NMS besteht im Erkennen und Melden von Fehlern. Zur Unterstützung der übrigen Aufgaben (Leistungs-, Konfigurations-, Sicherheits- und Abrechnungsmanagement) wird gefordert, dass alle gemessenen Daten zur Weiterverarbeitung in geeigneter Form abgelegt werden.

### 2.3.4.2 Heterogener Aufbau

Ziel eines NMS ist die betriebliche Aufrechterhaltung einer Netzwerkinfrastruktur. Heutige Netzwerke bestehen aus einem Verbund von vielen Rechnern unterschiedlicher Architektur und Betriebssysteme. Diese hochgradig heterogene Umgebung darf also keine Einschränkung für das NMS darstellen.

*Forderung 8:*

Ein NMS muss sowohl vom Aufbau als auch von der Funktion mit einer beliebig heterogenen Umgebung zurechtkommen.

### 2.3.4.3 Erweiterbarkeit

Um der Heterogenitätsbedingung zu genügen, muss ein NMS einfach erweiterbar sein, um beliebige Ressourcen an wechselnde Bedingungen zu koppeln (neue CPUs, zukünftige Betriebssysteme, ...). Sinnvoll erscheint somit nicht nur eine Trennung in der Funktionalität Manager/Agent, sondern vielmehr auch eine völlige Abstraktion von zu

überwachenden Eigenschaften. Ein modernes NMS sollte daher vollständig modular aufgebaut sein und durch beliebige Erweiterung einfach und kostengünstig angepasst werden können. Solche Erweiterungen werden gemeinhin als „Plug-Ins“ bezeichnet. Die eigentliche Programmlogik eines NMS kann sich somit auf das korrekte Abarbeiten dieser Plug-Ins konzentrieren, was die Komplexität und Fehleranfälligkeit des NMS wesentlich verringert. Idealerweise sollten solche Plug-Ins zur Laufzeit nachgeladen werden können.

*Forderung 9:*

Ein NMS muss modular im Sinne von Erweiterungen aufgebaut sein, so dass auf Änderungen an der Infrastruktur einfach durch Hinzufügen oder Entfernen eines Plug-Ins reagiert werden kann. Ebenso ist es dann möglich das NMS durch beliebige weitere Werkzeuge zu erweitern.

#### **2.3.4.4 Abbildung der organisatorischen Struktur**

Allgemein bildet die IT-Abteilung einen Teil der organisatorischen Struktur eines Unternehmens. In den wenigsten Fällen ist diese Abteilung aber flach strukturiert, vielmehr werden auch hier unterschiedliche Rollen anzutreffen sein.

Diese Struktur sollte möglichst natürlich auf das NMS übertragbar sein. Beispielsweise sollten bei kleineren Problemen nur die für die jeweilige Ressource zuständigen Mitarbeiter informiert werden (Fehlerzustand *Warnung*), während bei größeren Teilausfällen auch der zuständige Abteilungsleiter informiert werden sollte. In demselben Maße müssen aber auch Rechte und Rollen vergeben werden können, um geeignete Gegenmaßnahmen einzuleiten (nicht jeder Mitarbeiter sollte jede beliebige Information über die Infrastruktur einsehen können).

Unabhängig von den bisherigen Forderungen muss jedes NMS in der Lage sein, Benachrichtigungen (Meldungen) zu versenden. Dabei sollten eine Vielzahl an Kommunikationskanälen unterstützt werden. In der Regel sind dies E-Mail (ist aber wiederum abhängig von der Infrastruktur und damit fehleranfällig), SMS, Pager, Telefonanrufe, separate Alarmkanäle aber auch ein Trouble-Ticket-System.

Wünschenswert wäre eine weitere Verfeinerung der Benachrichtigungsmöglichkeiten. Es wird gefordert das unterschiedliche Parameter wie die Art der Benachrichtigung, Intervall, Ziel und Dringlichkeit abhängig von der Art des Fehlers (fehlerzustandsabhängig), von der aktuellen Uhrzeit/Datum (zeitabhängig) und der fehlerhaften Komponente (komponentenabhängig) individuell einstellbar sein müssen. Dabei sollten die definierbaren Abhängigkeiten (implizite wie explizite) auch auf die Benachrichtigungen angewandt werden, um unnötige Fehl- bzw. Folgefehleralarme zu verhindern.

Grundsätzlich müssen sich die Abhängigkeiten der organisatorischen Struktur auch in den Abhängigkeiten des Benachrichtigungssystems widerspiegeln.

*Forderung 10:*

Die organisatorische Struktur eines Unternehmens sollte sich möglichst natürlich auf das NMS übertragen lassen. Individuelle Rechte sollten für jeweilige Rollen vergeben werden können. Das Benachrichtigungssystem muss Teil der Umsetzung der organisatorischen Struktur sein.

Mit den Forderungen 1 bis 10 wurden Regeln definiert, anhand deren Systeme aus dem Bereich System- und Netzwerkmanagement verglichen werden können.

## 2.4 Weitere Kriterien

Neben den genannten zehn Hauptanforderungen können noch einige nebensächliche Kriterien bei der Beurteilung eines System genannt werden:

- Kosten für die Lizenz der Software
- Kosten für zusätzliche Hardware
- Kosten/Aufwand für die Wartung der Software (Wartungsverträge?)
- Kosten/Aufwand für die Schulung.

Diese Aufzählung ist sicherlich nicht vollständig, gibt aber einen ersten Anhaltspunkt, welche zusätzlichen Faktoren zu berücksichtigen sind. Dabei bleibt abzuwägen, wie diese Faktoren zu gewichten sind.

## 2.5 Zusammenfassung

Eine tabellarische Übersicht fasst die Anforderungen zusammen:

**Tabelle 2 Zusammenfassung der Anforderungen an ein NMS**

1	Datenspeicherung
2	Fehlerzustände
3	Manager und Agenten
4	Strukturierung
5	Abhängigkeiten
6	Protokolle
7	Fehlermanagement
8	Heterogener Aufbau
9	Erweiterbarkeit
10	Abbildung der organisatorischen Struktur
11	Kosten, Einführungsaufwand, Schulungsaufwand

## 3 Ist-Szenario: BigBrother

Im folgenden Kapiteln wird zunächst das bestehende System BigBrother den Anforderungen aus Kapitel 2 gegenübergestellt, und dessen Nachteile erläutert (Schwachstellenanalyse). Abschließend werden mögliche Nachfolgeprodukte vorgestellt und eines für die Evaluation im nächsten Kapitel ausgewählt.

### 3.1 Einführung

Zur Überwachung der Netzwerk- und Systemkomponenten setzt die ATIS derzeit das von der Firma *Quest Software* in zwei Varianten angebotene Softwaresystem BigBrother [30] ein. Während die Version BigBrother Professional Edition kostenpflichtig ist, kann man die zweite Variante im Rahmen eines Community Projekts kostenfrei beziehen, bei letzterer entfällt dafür der von *Quest Software* angebotene technische Support. BigBrother wurde zur reinen Überwachung (Monitoring, siehe 2.3.4.1) von Netzwerk- und Systemkomponenten entwickelt (Network- and Systems Monitor, NSM). Ein (teil-) automatisierter Steuerungseingriff (Controlling) im Fehlerfall ist nicht vorgesehen.

### 3.2 Aufbau und Funktionsweise

Nachdem im zweiten Kapitel die im Netzwerkmanagement üblichen strukturellen Rollen Manager/Agent eingeführt wurden, sollen diese nun mit der Implementierung des Produktes BigBrother in Verbindung gebracht werden.

Der Hersteller bewirbt das System als Klienten/Server-Lösung, bei dem der Server (BigBrother-Server) die Darstellung der von den BigBrother-Klienten gesammelten Information übernimmt.

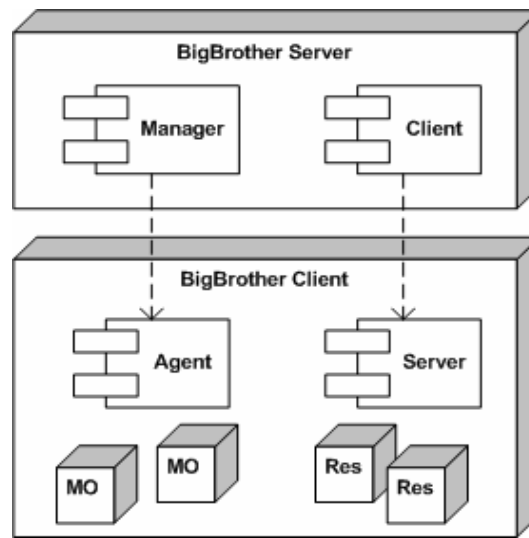
Die Anwendung BigBrother ist eine Sammlung von Shell-Skripten, wobei sowohl der BigBrother-Server als auch die BigBrother-Klienten auf verschiedenen unix-artigen Betriebssystemen wie auch auf der Microsoft Windows Plattform laufen können. Die BigBrother-Klienten werden auf den jeweiligen Hostsystemen eines Netzwerkes ausgeführt - den Managed Nodes (beispielsweise auf einem Router, Server, Firewall). Damit BigBrother-Klient und BigBrother-Server zusammenarbeiten, müssen bei der Installation zunächst jeweils generelle Einstellungen vorgenommen werden, wie beispielsweise die Festlegung der Portnummern zur Kommunikation. BigBrother verwendet hierzu standardmäßig den Port 1984.

Durch die Verwendung einer Skriptsprache sind sowohl die Klienten- wie auch die Serversoftware modular und erweiterbar aufgebaut, speziellere Monitoraufgaben lassen sich somit durch Hinzufügen eines angepassten Plugins erledigen. Dabei können beliebige Skriptsprachen verwendet werden, solange auf dem ausführenden System der passende Interpreter installiert ist. Daher hat sich auch seit dem Erscheinen der Software 1996 mittlerweile eine eigenständige Community entwickelt, die viele Plugins für spezielle Anwendungsfälle entwickelt hat und kostenfrei/quelloffen zur Verfügung stellt. Diese Tatsachen erfüllen insbesondere die Forderungen nach Erweiterbarkeit und Heterogenität (2.3.4.3, 2.3.4.2)

Bei der funktionalen Beschreibung der beiden Komponenten fällt jedoch auf, dass im Vergleich mit der im Netzmanagement etablierten Rollenverteilung (Manager/Agent) und den damit verbundenen Zuordnungen  $\text{Manager} \approx \text{Klient}$  und  $\text{Agent} \approx \text{Server}$ , die

vom Hersteller angegebene Definition der Klienten/Server-Beziehung nicht vereinbar ist. Schließlich hat diejenige Komponente mit der Rolle des Agenten eine klassische Serverfunktionalität, da sie dem Manager Information (durch ein Kommunikationsprotokoll) zur Verfügung stellt. Der Manager wiederum kann diese Daten abrufen und sammeln (klassische Klientenfunktionalität), um sie dann dem Benutzer (i.d.R. ein Netzadministrator) bereitzustellen. Erst in dieser Kommunikationsbeziehung zum Benutzer hin kommt dem BigBrother-Server tatsächlich die Serverrolle zu, der Benutzer (Webinterface) agiert dann als Client.

**Abbildung 6** verdeutlicht diesen Zusammenhang.



**Abbildung 6 Aufbau von BigBrother**

Weiterhin fällt auf, dass in der Terminologie des Herstellers von BigBrother nicht nur die Begriffe Manager/Agent nicht verwendet werden, sondern auch die Tatsache, dass die im Netzwerkmanagement bekannten Konzepte wie Managed Objects, Management Information Base und Management Protokoll keine Beachtung finden. Wie in diesem Abschnitt noch gezeigt werden wird, geschieht der Zugriff auf die gewünschte Information hier direkt und unmittelbar, eine abstrakte Beschreibung, wo welche Information zu finden ist, wie sie beispielsweise eine SNMP Management Information Base definiert, ist bei BigBrother nicht vorgesehen. Daher reduziert sich das eigentliche Kommunikationsprotokoll von BigBrother auf das Abfragen der Ergebnisse der Testfunktionen. Die Forderung unter 2.3.3.1 nach einem offenen Kommunikationsprotokoll wird hier also nur teilweise erfüllt.

Im Folgenden werden nun die Begriffe Agent (für die BigBrother-Klienten) und Manager (für den BigBrother-Server) verwendet.

Um BigBrother den (eigenen) Erfordernissen anzupassen ist die Justierung der Agenten und des Managers notwendig – dieses sind spezielle Einstellungen. Es ist festzulegen wie welcher Dienst auf einem Managed Node überwacht werden soll (oder überhaupt überwacht werden kann) und was als Fehlerzustand eines Dienstes definiert werden soll.

In diesem Zusammenhang sei hier auf die unterschiedliche Verwendung des Begriffes „Dienst“ hingewiesen. Im Kontext von BigBrother im Speziellen aber auch im

Netzwerkmanagementbereich im Allgemeinen wird der Begriff „Dienst“ für eine überwachte Eigenschaft verwendet und hängt nicht mit der klassischen Definition des Begriffes Dienst, d.h. aus Sicht eines Nutzers, zusammen. Gemeint ist hier also vielmehr der Begriff „Systemdienst“.

### 3.3 Die Agenten

Eine Vielzahl unterschiedlicher Basistests wie CPU-Load oder Festplattenauslastung werden vorkonfiguriert mit der Installation mitgeliefert und müssen nur noch der entsprechenden Situation auf dem Zielhost angepasst werden (beispielsweise unterscheiden sich die Anzahl und physikalische Anordnung von Festplattenpartitionen auf unterschiedlichen Hosts voneinander). Zusätzlich lassen sich sog. Testfunktionen definieren. Die Möglichkeiten dieser Testfunktionen sind jedoch eher eingeschränkt. Normalerweise beschränkt sich der Test eines bestimmten Dienstes auf die Existenz einer Prozessinstanz durch das Betriebssystem (beispielsweise durch eine Überprüfung mittels „if [ ps fax | grep Prozessname ]; then ...“) und nicht nach der Beurteilung, ob eine Dienstanfrage qualitativ das gewünschte Ergebnis liefert.

Beispielsweise könnte damit eine Situation eintreten, dass ein überwachter Prozess zwar vorhanden ist und dementsprechend keine Fehlermeldung generiert wird, der Prozess aber trotzdem nicht richtig funktioniert. Eine solche Situation kann die Fehlereingrenzung erheblich erschweren.

Die Fehlerkriterien für die Tests sind entweder Schwellwertangaben, welche gültige und ungültige (fehlerhafte) Wertebereiche von einander trennen, oder Soll-Ist-Vergleiche. Im Falle eines Fehlers generieren die Agenten einen Alarmbericht und senden diesen direkt an den Manager. Da der Agent nur für das Sammeln von Information und für die Generierung der Fehlernachrichten zuständig ist, fällt die Programmlogik auf den teilweise beschränkten Ressourcen entsprechend auch relativ gering aus.

### 3.4 Der Manager

Dem Manager kommen komplexere Aufgaben zu. Er generiert im Falle eines festgestellten Fehlers abhängig von der eingestellten Benachrichtigungsrichtlinie eine Warnmeldung und stellt diese Information zum einen auf seiner Weboberfläche zur Verfügung bzw. sendet diese direkt an den oder die zuständigen Mitarbeiter. Eine solche Richtlinie kann beispielsweise auf der aktuellen Uhrzeit, auf der Art des Dienstes oder auf der Art der Fehlermeldung basieren. Warnmeldungen werden in der Regel per E-Mail versendet, es besteht jedoch auch die Möglichkeit, über ein SMS-Gateway die Pager der zuständigen Mitarbeiter zu erreichen.

Die Manager-Funktionalität wurde vom Hersteller in drei unabhängige Prozesse, die untereinander kommunizieren, aufgeteilt. Eine Möglichkeit, die drei Teilprozesse auf unterschiedlichen Hostrechnern auszuführen ist nicht vorgesehen, jedoch besitzt der Server die Möglichkeit, als ganzes auf verschiedenen Host ausgeführt zu werden, wodurch ein gewisses Maß an Redundanz erreicht werden kann. Die einzelnen Aufgaben des Managers lassen sich wie folgt beschreiben:

- Tests ausführen: Dieser Prozess stellt das Herzstück von BigBrother dar und lässt sich in die oben beschriebenen Agentenbasierten Tests (Tests, die vom

jeweiligen Agenten auf einem Hostsystem ausgeführt werden) und die agentenlosen Tests unterteilen. Agentenlose Tests sind die Tests die der Manager direkt durchführen kann, welche also nicht auf das Vorhandensein eines Agenten angewiesen sind, darunter fallen beispielsweise alle Netzwerkprotokolltests wie http, dns, ldap, ftp, ssh und ähnliche. Hier stellt der Manager unmittelbar entsprechende Anfragen an das überwachte System.

- Information darstellen: dieser Prozess nimmt die von den Klienten eingehenden Nachrichten entgegen, speichert diese zur Analyse ab (in so genannten *logfiles*) und stellt diese Information in Form einer Weboberfläche zur Verfügung. Durch die Speicherung der Information in lokalen logfiles kann auch gezielt nach einer bestimmten Information gesucht werden. Es bietet sich damit auch die Möglichkeit, diese Informationen durch weitere Werkzeuge grafisch aufzubereiten (z.B. RRD-Tools [18]).
- Rundrufe durchführen: Prozess, der anhand von erkannten Fehlern, die Benachrichtigung (E-Mail, Pager) durchführt. (siehe 2.3.4.4)

### 3.5 Beispiel Mailsystem

Nach dieser einführenden Beschreibung soll nun an einem konkreten Beispiel gezeigt werden, wie BigBrother in der Praxis eingesetzt wird. Dazu wird die im ersten Kapitel eingeführte Mail-Infrastruktur herangezogen.

Die folgenden, zum Mailsystem gehörenden, Hosts und die darauf laufenden angegebenen Dienste werden derzeit mit BB überwacht:

- irams1.ira.uni-karlsruhe.de (zentraler Mailserver, webmail, ssh Zugang)
  - Dienste: connection, cpu, disk, http, imaps, msgs, pop3s, procs, smtp, ssh
- iramx1.ira.uka.de (Mail-Exchanger, ssh Zugang)
  - Dienste: connection, cpu, disk, msgs, procs, smtp, ssh
- iramx2.ira.uka.de (Mail-Exchanger, ssh Zugang)
  - Dienste: connection, cpu, disk, msgs, procs, smtp, ssh
- iraldap.ira.uni-karlsruhe.de (Verzeichnisdienst)
  - Dienste: connection, cpu, disk, http, msgs, procs, smtp, ssh

Stellvertretend für einen beliebigen Institutsmailserver werden hier die beiden für die studentischen Mailkonten zuständigen Server im Rechnerpool für die Studenten herangezogen:

- i08fs1.ira.uka.de (webmail Interface, ssh Zugang)
  - Dienste: connection, cpu, disk, http, msgs, procs, ssh
- i08fs2.ira.uka.de (studentischer Mailserver, ssh Zugang)
  - Dienste: connection, cpu, disk, imaps, msgs, pop3s, procs, smtp, ssh

Während die Überwachung der Dienste connection (Netzwerkping), http, imaps, pop3s, smtp und ssh Beispiele für die weiter oben beschriebenen agentenlosen Tests darstellen, kann die Information über cpu-load oder diskpace nur bezogen werden, wenn die



entsprechenden Test direkt auf den Klienten ausgeführt werden. Für die Information zu procs wird ein Prozessbaumabbild mit einer festgelegten Konfiguration verglichen, die Auftrittshäufigkeit eines Prozesses mit dem definierten Wert verglichen und im Falle einer Soll-Ist Abweichung ein Alarm generiert.

Beispielsweise würde die Definition dieser Tests für den studentischen Mailserver wie folgt aussehen:

```
localhost :: crond :  
localhost :: ntpd;=1 :  
localhost : : sshd  
localhost : : nfsd  
localhost : : xinetd  
localhost : : exim  
localhost : : smbd  
localhost : : nmbd  
localhost : : pxe;=1  
localhost : : dhcpd  
localhost : : nsrexecd;>=2
```

Die Syntax lässt sich beschreiben durch: *Hostname::Prozessname[;Mächtigkeit]*

In diesem Beispiel werden alle Tests auf dem lokalen Rechner ausgeführt (localhost). Die Namen der Dienste sind soweit selbsterklärend. Auffallend ist hier jedoch, dass bei vielen Diensten die Angabe über die Mächtigkeit fehlt. In diesem Falle wird der Standardwert „>=1“ verwendet (entspricht der Aussage mindestens eins). Möglich sind auch exklusive Aussagen (>1, mehr als eins), bzw. genaue Angaben (=1, genau eins).

Bietet dieses Vorgehen den Vorteil sehr flexibel zu sein (einfach den gewünschten Prozessname mit geforderter Auftrittshäufigkeit festlegen), so lassen sich aus diesen Informationen aber noch keine Schlussfolgerungen hinsichtlich der Güte des Mail-Dienstes aus Sicht des Nutzers ziehen (keinerlei Aussage über die Qualität der Antwort / Ausführungsdauer einer Dienstanfrage). Dennoch stellen diese Tests eine notwendige, grundsätzliche Verfügbarkeitsprüfung dar, Trivialerweise gilt hier der Zusammenhang:

*[Anzahl(Prozessinstanzen-Ist) < Anzahl(Prozessinstanzen-Soll)] ⇒ Mail-Dienst nicht nutzbar*

Ein großer Nachteil stellt die Tatsache dar, dass keine Abhängigkeiten definiert werden können. Gemessene Werte einer Ressource sind nur dieser zugeordnet, es besteht keine Möglichkeit, Komponenten, und damit auch die gemessenen Werte dieser Komponenten, zu gruppieren, um dadurch eine Aussage über die Güte eines (verteilten) Dienstes zu erreichen, dem diese Ressource zugeordnet ist. Eine Möglichkeit, den abstrakten Begriff „Mail-Dienst“ durch die Erfassung der Messwerte der beteiligten Ressourcen zu qualifizieren fehlt somit (Forderungen 2.3.2.2 und 2.3.2.3 werden nicht erfüllt).

### 3.6 Speicherung, Darstellung der gesammelten Information

Zentraler Bedeutung kommt der Speicherung und Darstellung der von den Agenten gesammelten Information zu. Nunmehr soll abschließend geklärt werden, wie BigBrother die von den Agenten gesammelte Information ablegt und diese darstellt.

Die Messwerte werden vom Managerprozess gesammelt und lokal in Textfiles abgelegt. Die Speicherung dieser Daten orientiert sich jedoch an keinerlei Standard für Informationsmodelle wie er beispielsweise durch CIM [11] (*Common Information Model*) beschrieben wird. So kann man zwar durch den Einsatz von Systemtools an gewünschte Information aus diesen Log-Files kommen, jedoch stehen diese Daten in keinem Zusammenhang mehr zu den überwachten Eigenschaften von diesen Hosts (Forderung 2.3.1.1 wird nicht erfüllt).

Der Eintrag in einem Log-File sieht beispielsweise folgendermaßen aus:

```
status irams2.conn green Sun Jan 21 14:16:36 CET 2007 Connection OK
PING 141.3.10.82 (141.3.10.82) 56(84) bytes of data.
64 bytes from 141.3.10.82: icmp_seq=0 ttl=63 time=0.311 ms
```

```
--- 141.3.10.82 ping statistics ---
```

```
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.311/0.311/0.311/0.000 ms, pipe 2
```

Aus diesen Informationen heraus generiert der Manager nun die Information für die Darstellung auf dem Webinterface. Im Folgenden werden nun einige Screenshots von diesem Webinterface präsentiert.

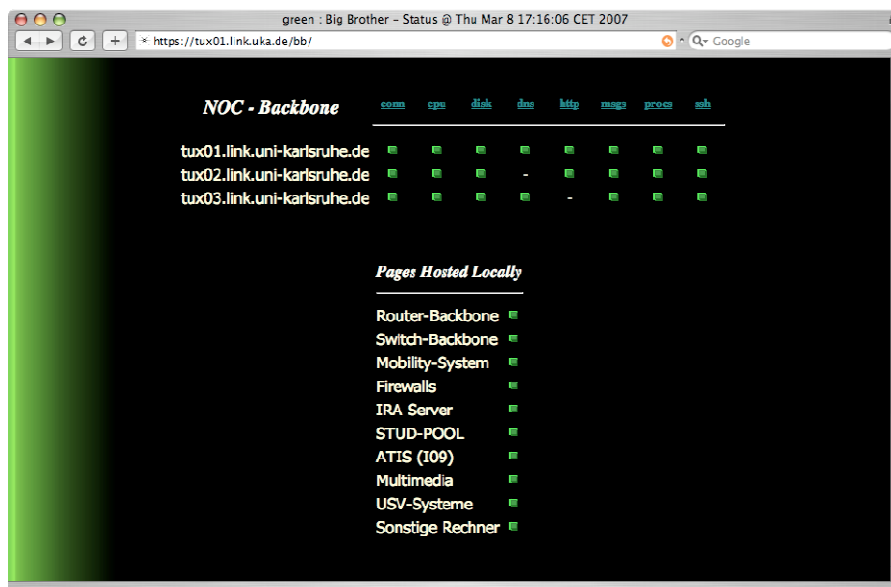
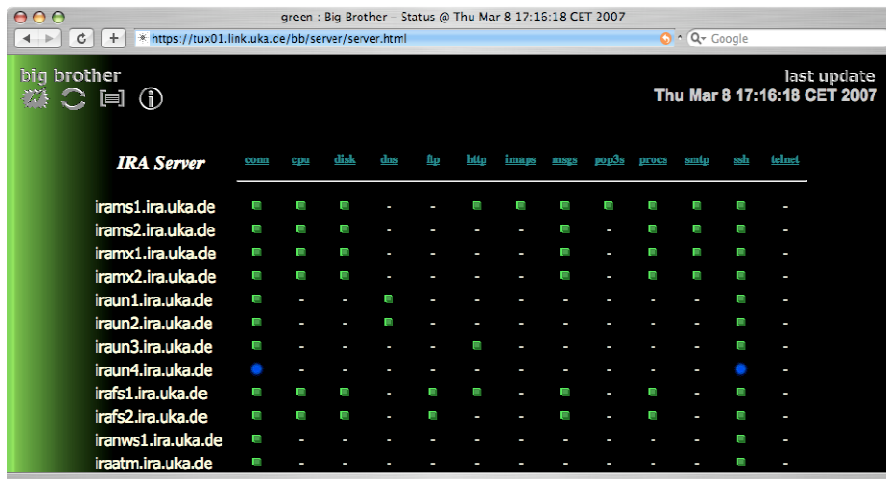


Abbildung 7 Einstiegseite BigBrother

Das Mailsystem verbirgt sich unter „IRA-Server“



The screenshot shows a web browser window with the URL `https://tux01.link.uka.de/bb/server/server.html`. The page title is "big brother" and the status is "green". The last update is "Thu Mar 8 17:16:18 CET 2007". The page displays a table of servers under the heading "IRA Server". The table has columns for various services: conn, cpa, diald, alus, ftp, http, images, ansgs, pop3s, ircws, snailp, ssh, and telnet. Each row represents a server, and each cell contains a green square indicating a successful connection, except for "iraun4.ira.uka.de" which has a blue square.

IRA Server	conn	cpa	diald	alus	ftp	http	images	ansgs	pop3s	ircws	snailp	ssh	telnet
irams1.ira.uka.de	■	■	■	-	-	■	■	■	■	■	■	■	-
irams2.ira.uka.de	■	■	■	-	-	-	-	-	-	■	■	■	-
iramx1.ira.uka.de	■	■	■	-	-	-	-	-	■	-	■	■	-
iramx2.ira.uka.de	■	■	■	-	-	-	-	-	■	-	■	■	-
iraun1.ira.uka.de	■	-	-	-	-	-	-	-	-	-	-	-	-
iraun2.ira.uka.de	■	-	-	■	-	-	-	-	-	-	-	■	-
iraun3.ira.uka.de	■	-	-	-	-	■	-	-	-	-	-	■	-
iraun4.ira.uka.de	■	-	-	-	-	-	-	-	-	-	-	■	-
irafs1.ira.uka.de	■	■	■	-	-	■	-	■	-	■	■	■	-
irafs2.ira.uka.de	■	■	■	-	■	-	-	-	-	■	■	■	-
iranws1.ira.uka.de	■	-	-	-	-	-	-	-	-	-	-	■	-
iraatm.ira.uka.de	■	-	-	-	-	-	-	-	-	-	-	■	-

Abbildung 8 connection Details

Bei einem Klick auf „conn“ erhält man zusätzliche Information zu dem Ergebnis des letzten Tests:



The screenshot shows a web browser window with the URL `https://tux01.link.uka.de/bb/html/irams2.ira.uka.de.conn.html`. The page title is "big brother" and the status is "green". The last update is "Thu Mar 8 17:13:06 2007". The page displays the connection details for "irams2.ira.uka.de - conn".

```

irams2.ira.uka.de - conn
green Thu Mar 8 17:13:04 CET 2007 Connection OK

PING 141.3.10.82 (141.3.10.82) 56(84) bytes of data.
64 bytes from 141.3.10.82: icmp seq=0 ttl=63 time=1.37 ms

--- 141.3.10.82 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.372/1.372/1.372/0.000 ms, pipe 2

Status unchanged in 7.32 days
Status message received from 192.168.1.250
  
```

Abbildung 9 Details für den Service connection

Über die Weboberfläche lässt sich auch die Historie eines Wertes abrufen, oder ein Report über einen gewissen Zeitraum erstellen (Verfügbarkeit in Prozent).

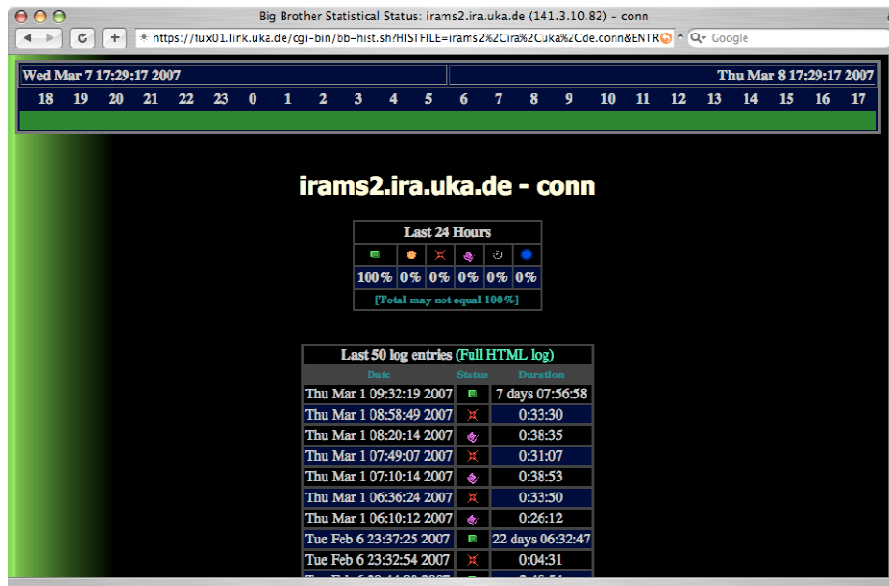


Abbildung 10 Historie für den Service connection

Durch das Webinterface besteht die Möglichkeit, prinzipiell von jedem Rechner mit Netzwerkverbindung vom BigBrother Server die gewünschte Managementinformationen abzurufen. Zum einen stellt dies einen Vorteil dar (Ortsunabhängigkeit beim Zugriff auf die Daten), zum anderen muss diese Information auch vor unbefugtem Missbrauch geschützt werden.

Das Webinterface bietet jedoch keinerlei Möglichkeiten, die Zugriffsrechte auf diese Informationen auf bestehende organisatorische Strukturen eines Unternehmens abzubilden. Man ist hierbei auf die vom benutzten Webserver angebotenen Authentifikationsmethoden angewiesen, um ein Mindestmaß an Vertraulichkeit zu gewährleisten (Forderung 2.3.4.4 wird nicht erfüllt)

### 3.7 Zusammenfassung BigBrother

Wie in diesem Kapitel gezeigt wurde, hat der Administrator mit BigBrother ein einfaches Werkzeug zur Hand, um Netzwerk- und Systemkomponenten zu überwachen (reines Monitoring). Es lassen sich verschiedene Systemprozesse und deren zu überwachende Eigenschaften, abhängig vom jeweiligen Hostsystem, angeben. Für diese Dienste werden, abhängig von einstellbaren Schwellwerten, im Fehlerfall Alarme generiert und versendet. Neben dem passiven agentenlosen Test von reinen Netzwerkdiensten bietet die Software die Möglichkeit spezielle angepasste Tests auszuführen, um eine größere Abdeckung bei der Überwachung der benutzten Dienste zu erreichen. Die Möglichkeiten hier sind jedoch eher eingeschränkt; es gibt keine Mechanismen, um Messwerte einheitlich zu erfassen und in Beziehung zueinander zu setzen.

Die zu überwachenden Ressourcen können nicht hierarchisch strukturiert werden. Es besteht zwar die Möglichkeit verschiedene Hosts zu gruppieren, jedoch kann weder diese Gruppierung rekursiv fortgesetzt werden (eine Hostgruppe kann keine weiteren Hostgruppen enthalten), noch können Systemprozesse (die zu überwachenden „Dienste“) gruppiert werden (zu Servicegruppen).

Weiterhin orientiert sich der Aufbau der Software an keinerlei standardisiertem Informationsmodell.

Der Eingriff im Fehlerfall (Controlling) erfolgt ausschließlich manuell über den zuständigen Administrator, ein automatisierter Handlungseingriff von BigBrother ist nicht möglich (reines Monitoring). Wichtiger Bestandteil eines NMS kommt der Speicherung und Darstellung der gesammelten Informationen zu. An dieser Stelle bietet BigBrother keine Möglichkeit, diese Information strukturiert abzulegen, um darauf später wieder zu zugreifen. Die Schnittstelle zum Benutzer wird durch ein Webinterface realisiert; Möglichkeiten, die organisatorische Struktur in den Monitorprozess abzubilden, existieren nicht.

Die Nachteile von BigBrother (Architektur, fehlende Standardisierung, Instrumentierung abstrakter Dienste, Zugriffsrechte) überwiegen die Vorteile (einfach zu verstehen, erweiterbar) bei weitem. Ein dienstorientiertes Management ist hiermit de facto nicht zu realisieren.

## 3.8 Alternativen zu BigBrother

An dieser Stelle sei auf die Studienarbeit von Thorsten Tüllmann verwiesen [12], dessen Zielsetzung neben der Beschreibung von ausgewählten Produkten auch ein Vergleich zum damals noch in der Version 1.2 vorliegenden Nagios war. Auf eine wiederholte Analyse dieser Produkte wird hier verzichtet, der Vollständigkeit halber aber die gewonnen Ergebnisse dieser Arbeit nochmals kurz zusammengefasst.

### 3.8.1 IBM Tivoli, openNMS

IBM Tivoli ist eine Sammlung von Softwareprodukten, um Informationssysteme zu verwalten. Die Möglichkeiten übersteigen die Anforderungen an ein NMS zur Infrastrukturüberwachung bei weitem, so ist es beispielsweise auch möglich, Software zu verteilen oder Datensicherungen durchzuführen[13].

Im Vorfeld der Arbeit [12] wurde IBM Tivoli von Mitarbeitern des Rechenzentrums evaluiert und aus verschiedenen Gründen für ungeeignet befunden. Hauptgründe sind wohl in den hohen Lizenzkosten, hoher Aufwand für Wartung und Pflege der eigentlichen Software und Komplexität des Produktes zu suchen, die einen Einsatz im Rechenzentrum nicht rechtfertigen.

OpenNMS [14] ist eine leistungsfähige OpenSource Lösung zum Überwachen von Netzinfrastrukturkomponenten. Zielsetzung des Projektes ist laut Angaben auf der Website, alle Aspekte des FCAPS Ressourcenmanagements umzusetzen. Momentan fokussiert sich die Entwicklung auf die Bereiche *Service Polling*, *Data Collection* und *Event Notification*.

OpenNMS wurde von Anfang an mit der Zielsetzung entwickelt, für den unternehmensweiten Einsatz betrieben werden zu können. So wurde auch SNMP als Standardprotokoll für die Kommunikation der zu überwachenden Infrastrukturressourcen verwendet.

### 3.8.2 Nagios

Hauptaufgabe von Nagios [15] ist es, Probleme und Fehler zu melden. Die Software ist vollständig modular aufgebaut und ist eher mit einem Framework vergleichbar. Die Programmlogik kann grob in drei Teile aufgeteilt werden: Abarbeitung der definierten Tests, Reaktion auf externe Ereignisse und die Bereitstellung einer Weboberfläche zur Visualisierung der aktuellen Objektzustände. Es besteht somit also die Möglichkeit,

zum einen regelmäßige Statusüberprüfungen durchzuführen (Polling), zum anderen externe Statusänderungen zu behandeln (Traps).

Nagios genügt allen Anforderungen, die im ersten Teil des zweiten Kapitels aufgestellt wurden. Durch den modularen Aufbau ist es flexibel erweiterbar und kommt mit beliebig heterogenen Umgebungen zurecht. In der Standardinstallation wird SNMP Unterstützung mitgeliefert, es können implizite und explizite Abhängigkeiten zwischen Hosts und Services auf unterschiedlichen Hosts definiert werden. Für jedes zu überwachende Objekt können getrennte Benachrichtigungsrichtlinien, bestehend aus Kontaktgruppen, Zeitintervallen, Benachrichtigungsaktionen und Eskalationsstufen angegeben werden.

Nagios unterscheidet grundlegend zwischen Hosts und Services. Unter einem Hosts versteht man eine physikalisch vorhandene Infrastrukturkomponente, beispielsweise ein Router, Switch oder ein Server. Hosts können gruppiert und über Hostgruppen aggregiert betrachtet werden.

Services im Kontext von Nagios sind nicht so klar definiert. Unter einem Service versteht man die Möglichkeit, sowohl Systemprozesse wie beispielsweise `exim`, zu modellieren, aber auch physikalische Ressourcen und ihre aktuelle Auslastung zu überwachen. Während Hosts also eher physikalische Entitäten modellieren, werden Services am besten durch die Zuordnung zu logischen Entitäten beschrieben. Ähnlich zu den Möglichkeiten bei Hosts, lassen sich auch beliebige Services zu Servicegruppen zusammenschließen. In der praktischen Anwendung wird davon auch reger Gebrauch gemacht, wie die beispielhafte Modellierung des Mailsystems im nächsten Kapitel zeigen wird.

Interessant ist der Ansatz, objektorientierte Funktionen nachzubilden. So besteht die Möglichkeit, für Host- und Servicebeschreibungen so genannte Templates anzulegen, um dann konkrete Objekte von diesen Templates abzuleiten. Die konkreten Objekte erben dann die bei den Templates definierten Eigenschaften.

### **3.9 Auswahl von Nagios für die Evaluation**

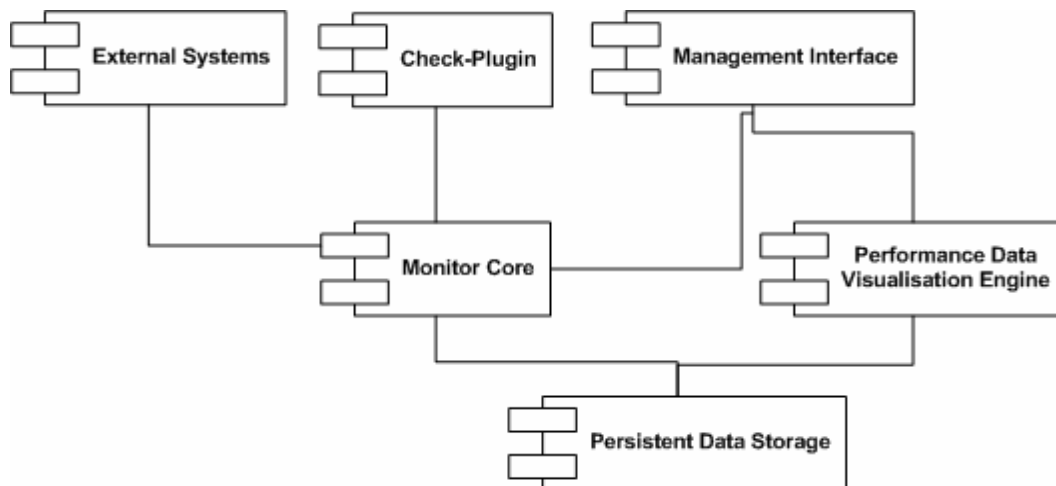
Wie dieses Kapitel gezeigt hat, genügt BigBrother vielen der gestellten Anforderungen nicht mehr. Aufgrund der Ergebnisse der Arbeit [12] und der stark weiter entwickelten Version von nagios (nun in der Version 3.0), wird nagios für eine Evaluation bezüglich der formulierten Anforderungen vorgeschlagen. Die Evaluation findet im folgenden Kapitel statt.

## 4 Evaluation Nagios (Version 3)

In diesem Kapitel wird nagios (Version 3.0) für den Einsatz in der ATIS evaluiert. Als Grundlage dafür dienen die im zweiten Kapitel abgeleiteten Anforderungen. Es wurde bereits gezeigt, dass BigBrother diese Anforderungen zum größten Teil nicht erfüllt.

### 4.1 Instanz einer Monitoring Architektur

Beschränkt man sich bei der im zweiten Kapitel eingeführten Managementarchitektur auf das reine Monitoring, kann das Modell wie in **Abbildung 11** gezeigt, folgendermaßen konkretisiert werden. (Instanziierung).



**Abbildung 11** Instanzen einer Monitoring Architektur

Das System besteht im Prinzip aus folgenden Komponenten: *Monitoring-Core* (zentrale Komponente), *Data Storage*, *Management Interface* und *Performance Data Visualisation*.

Zur Funktionsweise: Die Komponente Monitoring-Core ermittelt Messdaten unmittelbar selbstständig (synchron) durch aktives Ausführen von so genannten Checks oder erhält zu beliebigen Zeitpunkten Messdaten und/oder Ereignisse durch beliebige weitere Agenten (passiv, asynchron). Diese Agenten können wiederum beliebige weitere Systeme zur Verarbeitung von Ereignissen sein. Dadurch können zusätzliche, spezialisierte Systeme eingebunden werden.

Die gesammelten Daten werden persistent abgelegt und zur Weiterverarbeitung zur Verfügung gestellt. Dabei könnten hier sowohl einfache log-files wie auch standardisierte Datenbanken zur Anwendung kommen. Über ein Management Interface kann sowohl der aktuelle Zustand des Systems wie auch Performanceinformation über aggregierte Daten aus der Vergangenheit des Systems veranschaulicht werden. Weiterhin kann die aktuelle Konfiguration des Monitor-Systems betrachtet und im Idealfall auch geändert werden.

Leistungsdaten (Performance-Data) werden meist zur Anfragezeit generiert (*on-demand*). Dazu greift die Visualisierungseingine bei Bedarf auf die im Data-Storage abgelegten Leistungsdaten zu und erstellt die vom Benutzer gewünschte Information (Leistungsbericht, Balkengrafik oder ähnliches). Die Leistungsdaten werden direkt in das Managementinterface eingebunden, können aber auch unabhängig davon abgefragt werden.

## 4.2 Aufbau und Architektur von Nagios

### 4.2.1 Einleitung

Nagios (in der aktuellen Entwicklerversion 3.0) ist ein Opensource-Produkt zur Überwachung von Hostrechner und den damit verbundenen Systemdiensten. Eine gute Übersicht über den Funktionsumfang findet sich auf der Webseite [16] des Projekts wieder.

### 4.2.2 Umsetzung des Monitoringsystems

Nagios ist vollständig modular aufgebaut (Framework Charakter). Dadurch kann das im vorigen Abschnitt dargestellte Monitoring-System einfach umgesetzt werden. **Abbildung 13** verdeutlicht den Aufbau des Systems.

Der Nagios Core stellt die zentrale Monitoring-Anwendung dar. Messergebnisse können auf drei verschiedene Weisen ermittelt werden: Direktes Ausführen des Plugins durch den Nagios Core (synchron, aktiv), Anstoßen eines Plugin-Tests auf einem entfernten Host durch den *Nagios Remote Plugin Executor* (NRPE, ebenfalls synchron und aktiv) und Entgegennahme eines Messwertes von Dritt-Systemen durch den *Nagios Service Check Acceptor* (NSCA, asynchron, passiv). Bei der direkten Ausführung eines Plugins sind keine Grenzen gesetzt. Im Prinzip kann alles direkt überwacht werden, Bedingung ist, dass eine Konnektivität über das angebundene Netzwerk besteht. So kann beispielsweise auch sehr einfach eine Monitoring-Anwendung installiert werden, die vollständig auf SNMP basiert. Ein SNMP-Plugin wird bereits bei der Standardinstallation mitgeliefert.

Beim passiven Überwachen von Messwerten muss der entsprechende Test dem Core bei der Konfiguration mitgeteilt werden, anschließend werden nur Ereignisse verarbeitet, die der Core von Drittsystemen empfängt.

Die Kommunikation zwischen dem Monitoringcore und den einzelnen Komponenten lässt sich wiederum abhängig von der Art des Tests beschreiben: während beim direkten Ausführen eines Check-Plugins (beispielsweise ein http-check auf einen Webserver) zwischen dem Test-Plugin und dem Core keine Netzwerkkommunikation stattfindet (sehr wohl jedoch zwischen Test-Plugin und getestetem Service auf dem entsprechenden Zielhost), ist das Verhältnis beim Testen mittels NRPE genau entgegengesetzt. Hier findet in der Regel keine Netzwerkkommunikation zwischen Test-Plugin und dem zu testenden Dienst statt, jedoch wird ein offengelegtes Protokoll zwischen dem Core und dem entfernten NRPE benötigt. Nagios bietet dieses Protokoll, wenn auch in sehr einfacher Weise. Über einen Befehlsmechanismus schickt der Nagios-Core die Befehlsaufforderungen an den entfernten NRPE, der wiederum das Check-Ergebnis an den Nagios-Core übermittelt.

Das Kommunikationsprotokoll des NSCA lässt sich noch weiter reduzieren, da hier kein Befehlsmechanismus benötigt wird und lediglich Funktionalität zur Übermittlung des Testergebnisses implementiert.

Über ein Webinterface kann sowohl der aktuelle Systemzustand wie auch die Konfiguration von nagios überwacht werden, letztere kann hier jedoch (noch) nicht geändert werden. Zur Konfiguration einer Infrastruktur (Definition der zu Überwachenden Hosts und Services etc.) muss also entweder eine zusätzliche Software verwendet werden (beispielsweise fruity [17]) oder aber die Konfiguration wird direkt im System durch ändern der Konfigurationsdaten erledigt.



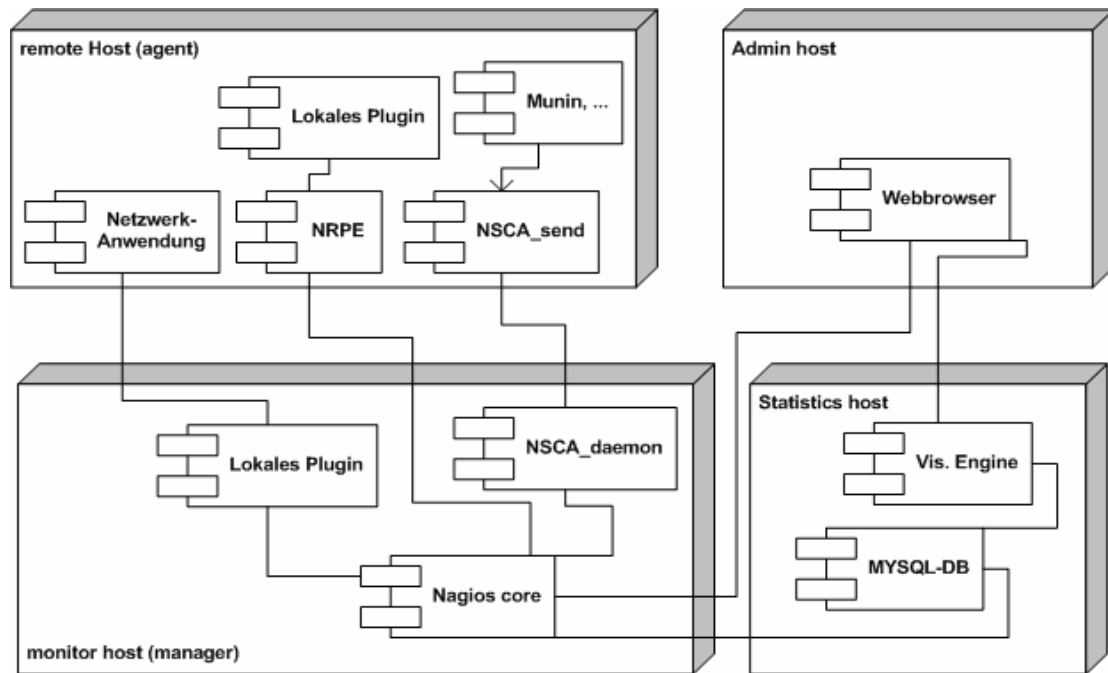


Abbildung 12 Monitoranwendung

Nagios bietet die Möglichkeit, Performance-Information unmittelbar bei Eintreffen eines Messergebnisses weiter zu verarbeiten. Dazu wurden die weit verbreiteten RRDTools[18] verwendet, die auch schon im bestehenden Netzwerkmanagement der ATIS zum Einsatz kommen. Bestand bei BigBrother noch nicht Möglichkeit, Messergebnisse mit den gesammelten Performancedaten zu kombinieren (BigBrother und die RRDTools sind zur Zeit noch zwei voneinander unabhängige Systeme), können die Performancedaten nun unmittelbar in die Weboberfläche von nagios integriert werden.

### 4.2.3 Architektur von nagios

Im Zentrum der Überwachungslogik von nagios steht der Begriff *Objekt*. Ein Objekt stellt eine formale Beschreibung einer Entität in nagios dar. Es wird eine Vielzahl verschiedener Objekttypen unterschieden. Diese mit allen Eigenschaften aufzuzählen würde den Rahmen dieser Arbeit bei weitem sprengen, für eine genaue Beschreibung sei auf [19] verwiesen.

Es lassen sich jedoch grundlegende Zusammenhänge grafisch darstellen. Die Formulierung dieser Zusammenhänge entspricht der Forderung, Information strukturiert darzustellen und abzulegen (Information Model, siehe 2.3.1).

Visual Paradigm for UML Community Edition [not for commercial use]

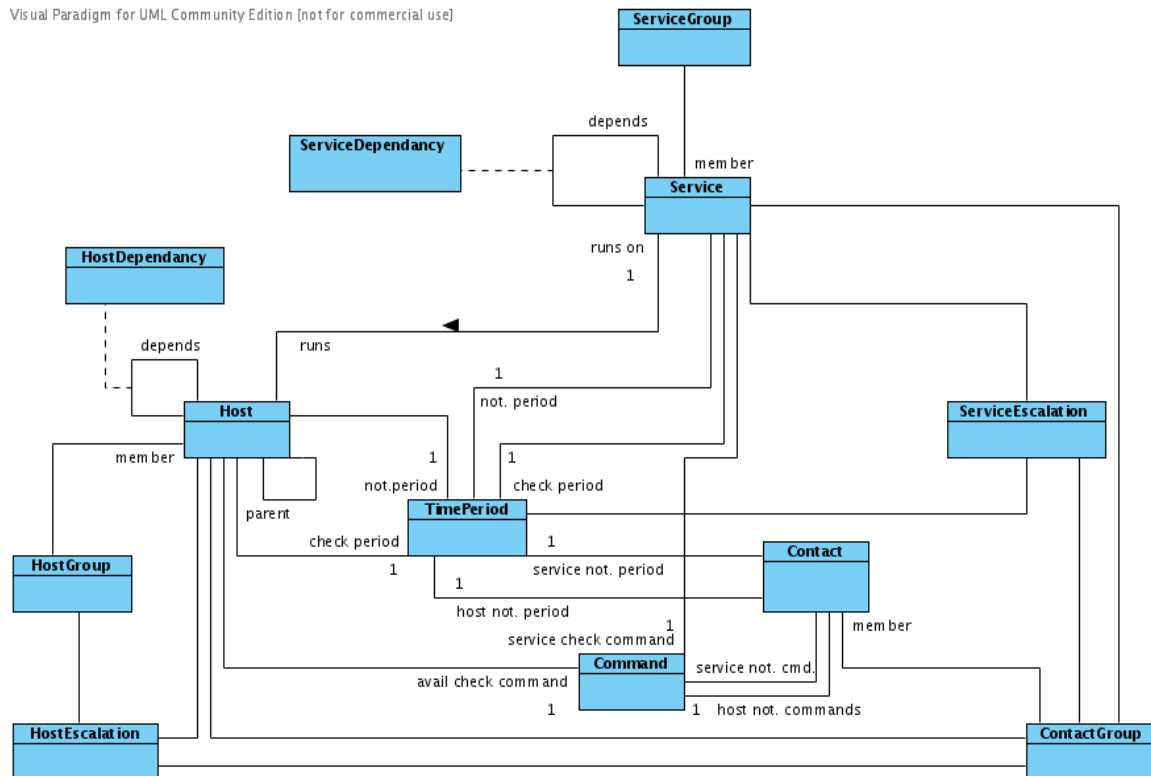


Abbildung 13 Zusammenhänge der Objekttypen - Klassendiagramm

**Host** – Zentrales Objekt zur Beschreibung einer physikalischen Entität, also ein Server, Router, Firewall, Accesspoint u.v.m. Ein Host wird über einen Hostnamen und eine IP-Adresse identifiziert.

**Hostgroup** – Hosts, die logisch zusammengehören, können in einer Hostgruppe zusammengefasst werden.

**Service** – Das Gegenstück zur Modellierung einer logischen Entität bildet das Objekt Service. Einem Service sind mindestens ein Name und ein Kommando, um diesen Service zu testen, zugeordnet.

**Servicegroup** – ebenso wie Hosts können auch Services zu einer Gruppe zusammengeschlossen werden. Dies stellt ein wichtiges Feature von nagios dar und ist einer der Hauptgründe, warum ein Nachfolgesystem für BigBrother gesucht wurde.

**Timeperiod** – Eine Timeperiod definiert Zeitintervalle, die wiederum zu fast allen Objekten zugeordnet werden können. Dadurch kann feingranular eingestellt werden, was genau zu welcher Zeit überwacht und wie dann abhängig von der Zeit reagiert werden soll

**Contact** – Ein Kontakt definiert eine Benachrichtigungsadresse, die im Problemfall erreicht werden soll, beispielsweise ein Systemadministrator

**Contactgroup** – Kontakte können gebündelt werden und so organisatorisch zusammengehörende Einheiten verbunden werden

**Command** – Ein Kommando definiert zu einer Aliasbezeichnung (Schnellzugriffsbezeichnung) ein Befehl oder eine Aktion. Dadurch werden Komplizierte Tests mit einer Vielzahl an Parametern hinter einem einfach zu verwendendem Kürzel verborgen.

**Hostdependancy, Servicedependancy** – Abhängigkeiten in der Infrastruktur können explizit formuliert werden. Durch diesen Mechanismus wird eine Infrastruktur so natürlich wie möglich in das Monitoringsystem abgebildet.

**Hostescalation, Servicescalation** – Durch Eskalationsstufen im Benachrichtigungsprozess können unnötige Frühalarne wie auch unwichtige Warnhinweise hinausgezögert werden, um bspw. Kosten für die Benachrichtigung auf ein Mobiltelefon per SMS zu senken.

### 4.3 Umsetzung Mailsystem

Im zweiten Kapitel wurde dargelegt, welche grundsätzlichen Zusammenhänge zwischen den Systemdiensten im Mailsystem bestehen. **Abbildung 14** verdeutlicht den Zusammenhang zwischen vorhandenen Objekten (Hosts, Systemdiensten) und deren Entsprechungen in der modellierten Infrastruktur in nagios.

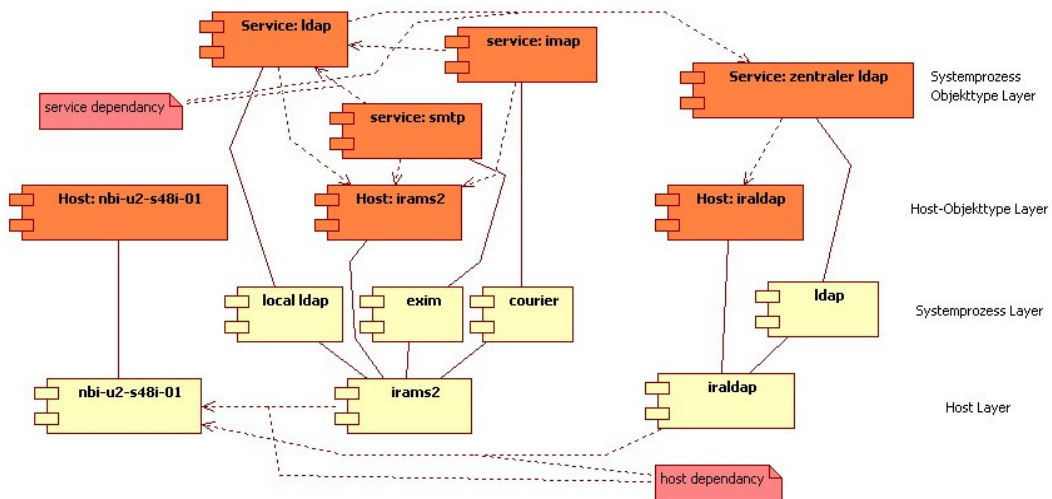


Abbildung 14 Zusammenhang zwischen Objekten und nagios Modellierung

Beispielhaft wird nun beschrieben, wie die unterschiedlichen Komponenten des Mailsystems mittels den weiter oben vorgestellten Objekttypen modelliert werden können.

#### 4.3.1 Host - Infrastrukturkomponente – Switch

Die Infrastrukturkomponenten werden nur auf Erreichbarkeit überprüft. Diesen Komponenten sind keine weiteren Systemdienste zugeordnet, die Konfiguration beschränkt sich somit auf Angabe des Namens, der IP-Adresse zur Erreichbarkeitsprüfung sowie einer Parent-Beziehung zur Definition der physikalischen Abhängigkeit von einer anderen Infrastrukturkomponente.

#### 4.3.2 Host - Infrastrukturkomponente – Firewall

Wie auch beim Switch wird dieser Komponente bisher kein Systemdienst zugeordnet. Da die Firewalls innerhalb der Netzwerkinfrastruktur aber die Rolle eines Routers einnehmen, wäre denkbar, in Zukunft Beschreibungen wie Routingtabellen o.ä. mit in die Netzwerküberwachung, auch im Hinblick auf ein dynamisches Routing, zu nehmen. Weiterhin werden auch hier Parent-Beziehungen zur Angabe von physikalischen Abhängigkeiten formuliert.

### 4.3.3 Host - Mailtransportserver – iramx1

Serversysteme unterscheiden sich von reinen Infrastrukturkomponenten durch die Zuordnung von Systemdiensten. Die Definition des Host-Systems gestaltet sich analog zur Konfiguration von Infrastrukturkomponenten, Parent-Beziehungen helfen auch hier, physikalische Abhängigkeiten zu modellieren.

Zur Umsetzung der Systemdienste dient der Service-Objekttyp. Dabei können beliebig viele Systemdienste einem Host zugeordnet werden. Am Beispiel der iramx1 findet man die Systemdienste Exim Mailqueue, cpu\_load, disk\_usage, ldap und smtp. Die Systemdienste ldap bzw. smtp werden direkt vom nagios core durch die entsprechenden Plugins check\_ldap bzw. check\_smtp geprüft. Bei den Diensten cpu\_load und disk\_usage kommt der NRPE zum Einsatz, da diese Information zwar auch mittels SNMP verfügbar ist, bei der bisherigen Umsetzung aber nicht zum Einsatz kam. Die Exim Mailqueue wird über den NSCA passiv überwacht, ein aktives Monitoring findet hier nicht statt.

### 4.3.4 Hostgruppen

Es lassen sich folgende Hostgruppen zur Aufteilung der physikalischen Gruppierungen unterscheiden: Firewalls, Infrastrukturhostgroup, Institutsmailserver, Server des zentralen Verzeichnisdiensts und zentrale Mailserver.

### 4.3.5 Servicegruppen

Im Rahmen dieser Arbeit wird auch untersucht, inwiefern ein NMS eine Orientierung bezüglich Dienstgüteunterstützung (*Quality of Service, QoS*) bieten. Nagios bietet zur Unterstützung dieser Vorgaben die Möglichkeit, Systemdienste zu Servicegruppen zusammen zu schließen, welche wiederum rekursiv zu weiteren Servicegruppen zusammenschlossen werden können.

Bei der Umsetzung des Mailsystems ließen sich folgende Servicegruppen identifizieren: Mail-Postfachdienste, Mail-Transportdienste (beide Teil-Dienste auch jeweils für Institutsmailserver), ldap-basierte Verzeichnisdienste sowie physikalische Ressourcen der beteiligten Systeme.

Mag diese Aufteilung auf den ersten Blick intuitiv erscheinen, kommt bei genauerer Betrachtung der Charakter des Mail-Dienstes zu Tage. Es lassen sich hier grob zwei Teildienste finden, die den beiden Dienstoperationen Mail-Lesen und Mail-Senden unmittelbar entsprechen. An dieser Stelle sei nochmals auf Kapitel 2 verwiesen, hier wird der Aufbau des Maildienstes detailliert dargelegt.

## 4.4 Datenspeicherung und Visualisierung

### 4.4.1 Datenspeicherung

Wichtige Forderungen an ein NMS werden in Bezug auf die Art und Weise gestellt, wie die gesammelten Mess- und Ereignis-, aber auch die Konfigurationsdaten, abgelegt werden, aber auch visualisiert werden.

Nagios bietet hierzu mehrere Möglichkeiten, die unterschiedlich zu bewerten sind. Eine Datenbankanbindung ist vorgesehen, so kann beispielsweise eine mysql-Datenbank eingesetzt werden, um Leistungsmessdaten strukturiert abzulegen. In der Standardinstallation werden sowohl Konfigurations- wie auch Messdaten in Textfiles

abgelegt. Auf diese Daten kann einfach zugegriffen werden, jedoch sind die Möglichkeiten zur Weiterverarbeitung eingeschränkt.

In der Umsetzung im Rahmen dieser Studienarbeit wurde zusätzlich die Anbindung an eine mysql-Datenbank konfiguriert. Dazu muss zur Grundinstallation von nagios ein so genanntes Event-Broker Module installiert werden, welches die Kommunizierung der ermittelten Messdaten (Leistungsinformation oder Ereignisdaten) zu einem separaten Daemonprozess vornimmt. Dieser separate Daemonprozess schreibt dann letztendlich die gewünschten Daten in die Datenbank. Durch die Aufteilung in zwei separate Module (Eventbroker und Daemonprozess) besteht die Möglichkeit, beliebige Informationsspeichersysteme mit nagios zu koppeln. Denkbar wäre beispielsweise auch die Integration eines CIM-konformen Datenspeichersystems [11].

Die nachfolgende Abbildung verdeutlicht den Zusammenhang.

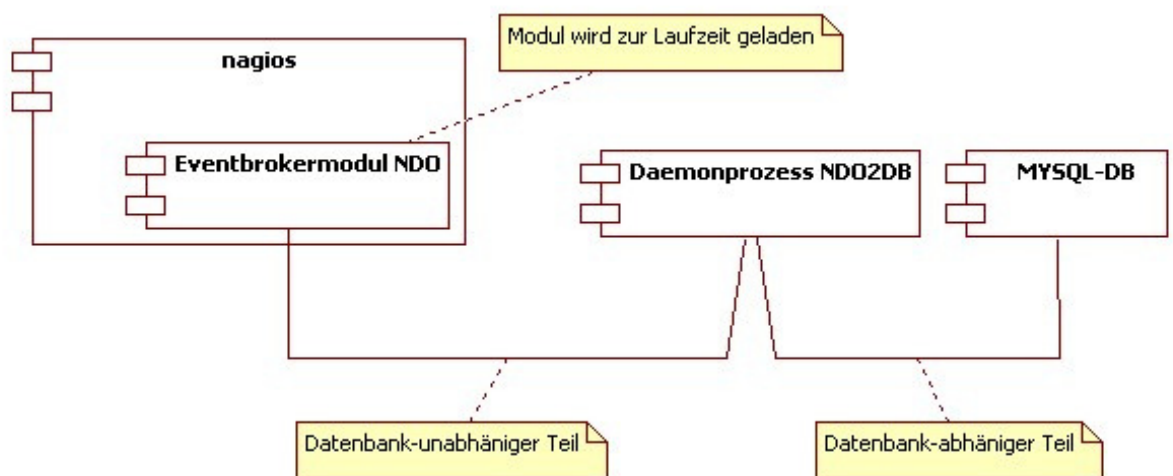


Abbildung 15 Integration NDO (Nagios Data Objects)

Ein vollständig beschriebenes Datenmodell ist auf der Website des Projekts erhältlich [20].

#### 4.4.2 Visualisierung

Zur Weitergehenden Beurteilung der Qualität einer Ressource sind Leistungsdiagramme ein wichtiges Hilfsmittel in der Arbeit eines Netzwerkadministrators, da sie die Diagnose von Trends erheblich erleichtern. Nagios kann die gesammelte Leistungsinformation eines checks ablegen und so weiteren Systemen ermöglichen, auf diese Daten zu zugreifen. Weiterhin können diese Leistungsdiagramme (und damit die entsprechende Leistungsinformation) direkt in die Oberfläche von nagios integriert werden. Abbildung 16 zeigt die Integration der RRD-Tools zur Ablage und Visualisierung der Leistungsinformation mit nagios.

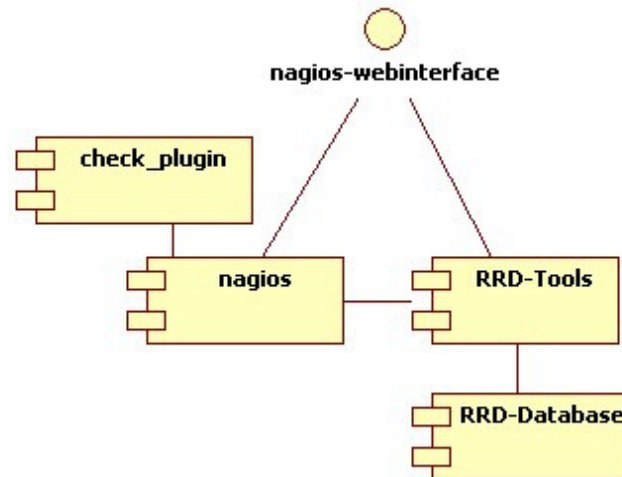


Abbildung 16 Integration RRD-Tools

## 4.5 Webinterface

In der Standardinstallation bietet nagios ein umfassendes und einfach zu bedienendes Webinterface. Ausgangspunkt für die Arbeit damit ist die „Taktische Übersicht“, welche in Kurzform Informationen über die zu überwachende Infrastruktur liefert.

The screenshot shows the Nagios web interface in Mozilla Firefox. The main content area is titled 'Tactical Monitoring Overview' and includes the following sections:

- Monitoring Performance:**
  - Service Check Execution Time: 0.11 / 1.32 / 0.521 sec
  - Service Check Latency: 0.33 / 4.89 / 2.696 sec
  - Host Check Execution Time: 0.11 / 1.00 / 0.376 sec
  - Host Check Latency: 0.01 / 5.30 / 1.842 sec
  - # Active Host / Service Checks: 42 / 50
  - # Passive Host / Service Checks: 0 / 2
- Network Outages:** 0 Outages
- Network Health:** Host Health: ██████████ Service Health: ██████████
- Hosts:**

1 Down	0 Unreachable	41 Up	0 Pending
--------	---------------	-------	-----------
- Services:**

1 Critical	1 Warning	0 Unknown	50 Ok	0 Pending
------------	-----------	-----------	-------	-----------
- Monitoring Features:**

Flap Detection	Notifications	Event Handlers	Active Checks	Passive Checks
Enabled All Services Enabled No Services Flapping All Hosts Enabled	Enabled All Services Enabled All Hosts Enabled	Enabled All Services Enabled All Hosts Enabled	Enabled 2 Services Disabled 1 Host Disabled	Enabled All Services Enabled All Hosts Enabled

Abbildung 17 Einstiegsseite - Taktische Übersicht

Übersichtsseiten für die Hosts geben einen schnellen Überblick über alle überwachte Hosts.

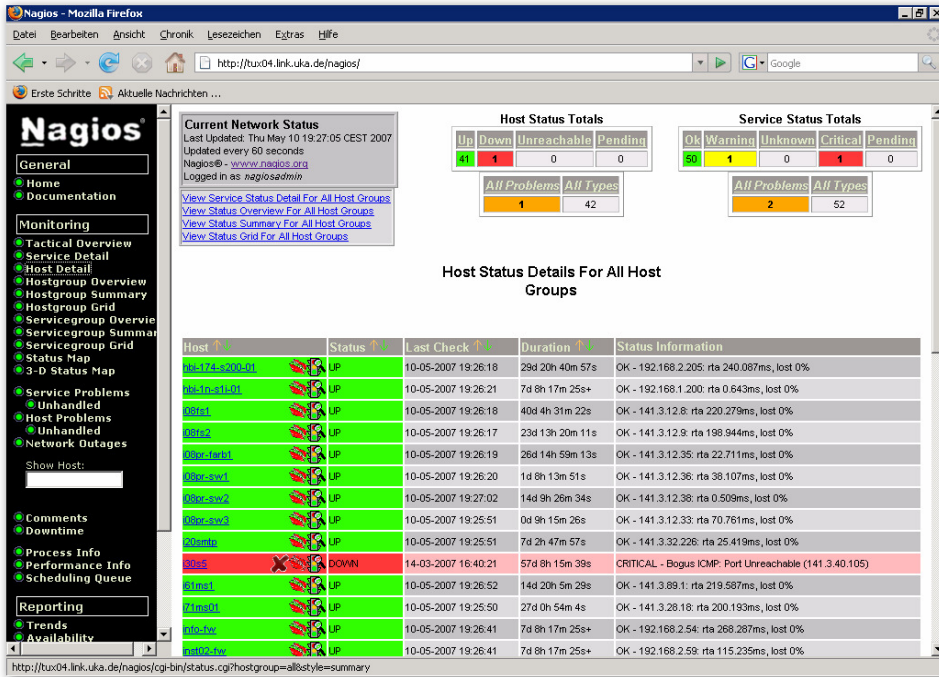


Abbildung 18 Hoststatus Übersichtsseite

Ebenso die Übersichtsseite über zu überwachende Systemdienste.

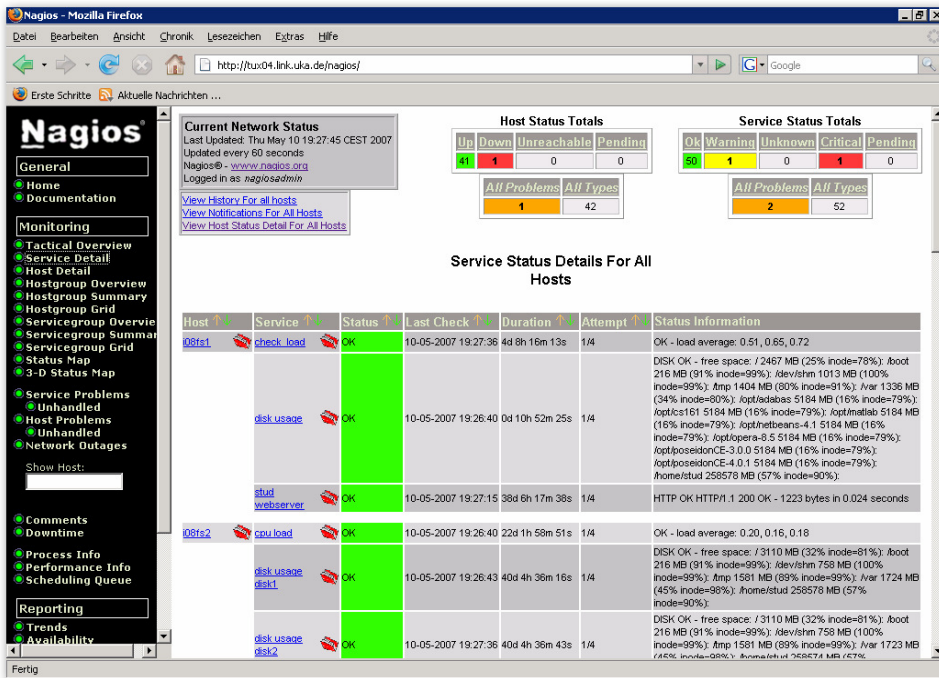


Abbildung 19 Servicestatus Übersichtsseite

Es ist möglich, die Service-Groups auch in der Visualisierung aggregiert zu betrachten.

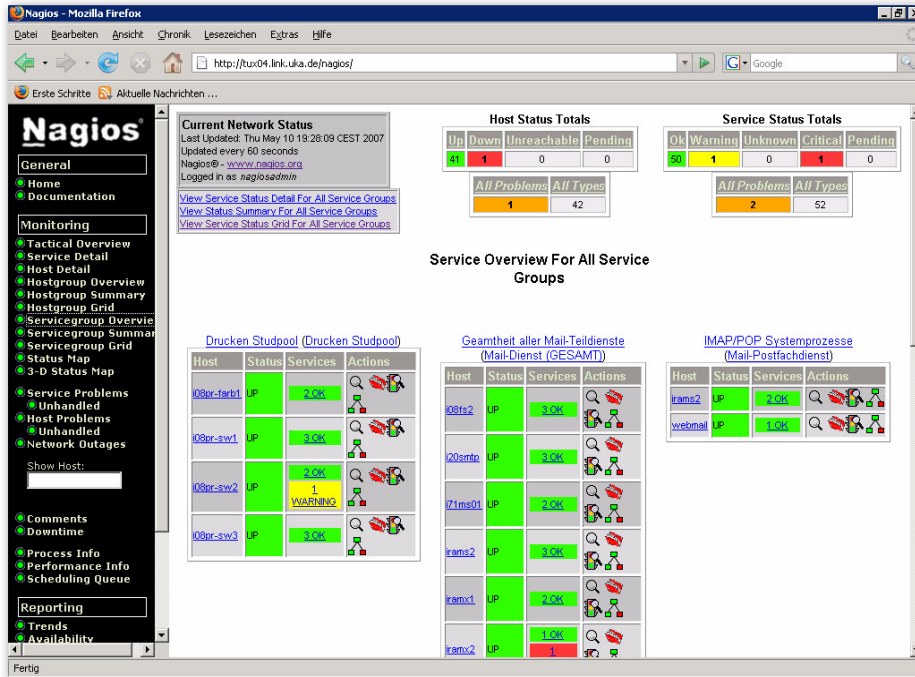


Abbildung 20 Servicegruppen Übersichtsseite

Die zu überwachende Infrastruktur kann grafisch visualisiert werden, um so beispielsweise einen Überblick über komplizierte Netzwerkwerke zu erhalten.

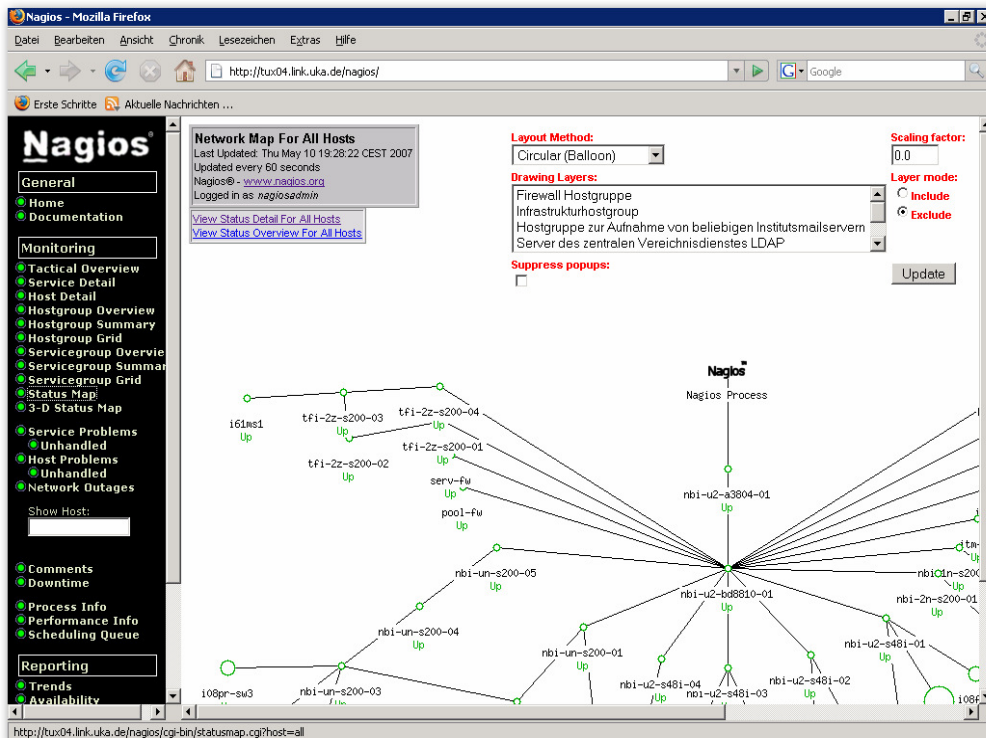


Abbildung 21 Visualisierung der Infrastruktur

Performance-Information kann direkt in die Oberfläche integriert werden. Dazu wurden die RRD-Tools so konfiguriert, dass bei Eintreffen einer Performanceinformation die Daten in die entsprechende RRD-Tabelle eingetragen werden. Ein Klick neben das



jeweilige entsprechende Objekt generiert zur Laufzeit ein Diagramm zur Darstellung von Leistungsinformation:

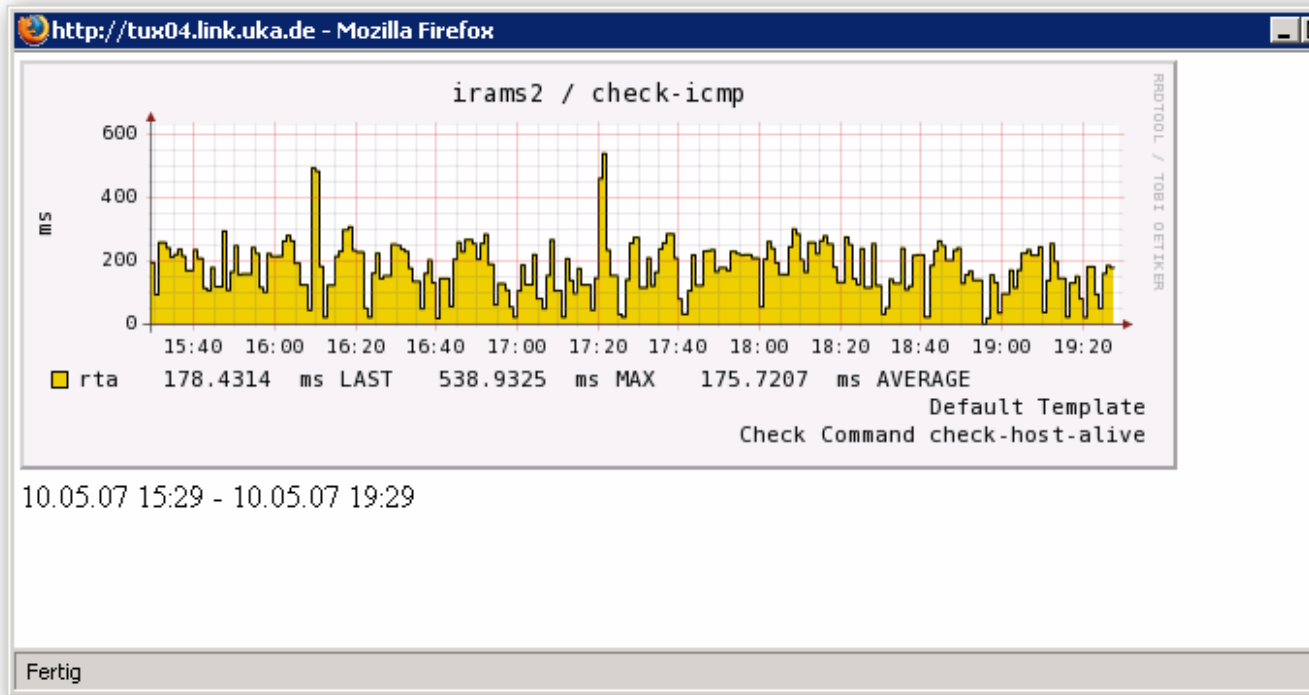


Abbildung 22 Eingebettete Leistungsinformation

### 4.6 Verwaltung von nagios

Um auch komplexe Szenarien zu überwachen, stellt sich die Frage, inwiefern der Administrator von nagios unterstützt wird, die zu überwachende Infrastruktur modellieren zu können. Da nagios frei verfügbar ist, haben sich mittlerweile verschiedene Ansätze entwickelt, die die Konfiguration von nagios und der zu überwachenden Infrastruktur vereinfachen sollen. Im Folgenden wird nun neben der einfachen Installation sowohl ein kooperierender (Webinterface fruity) wie auch ein integrierter Ansatz (Groundworks Opensource Monitor) vorgestellt. Abbildung 23 zeigt den Zusammenhang der drei unterschiedlichen Ansätze auf.

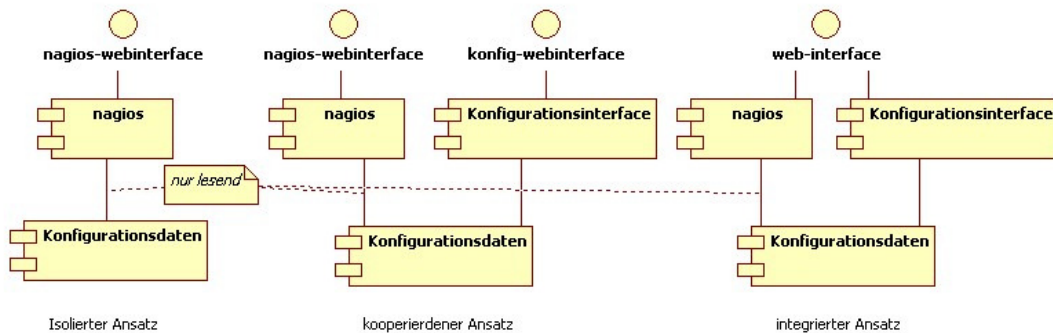


Abbildung 23 Vergleich: isolierter, kooperierender, integrierter Ansatz

### **4.6.1 Installation und Konfiguration**

Um nagios auszuführen wird im einfachsten Fall ein Standard Personalcomputer mit funktionierender Netzwerkunterstützung (TCP/IP) und unix-artigem Betriebssystem benötigt.

Bei der Testinstallation zur Umsetzung des Mailsystems stand ein Pentium/III mit ca. 1 GHz Rechenleistung und 1 GB Hauptspeicher zur Verfügung. Gehostet wird die Installation von FedoraCore 5. Die Anforderungen an die restlichen Systemkomponenten sind eher gering, jedoch sollte bedacht werden, dass ständiges Mitschneiden der Debugausgaben in logfiles schnell in den GB-Bereich führen kann.

Die Installation von nagios basiert auf der aktuellen Entwicklerversion 3.0a3. Obwohl diese Version gegenüber der letzten stabilen Version der 2.X-Reihe offiziell nicht als stabil gilt, wurde im Testbetrieb kein Absturz oder unerklärliches Fehlverhalten festgestellt. Es wurde eine Sourcecode-Installation mit selbstständiger Übersetzung und Anpassung des Quellcodes an das Zielsystem durchgeführt. Die Installation verlief dank automatischer Archivkonfiguration mittels configure, wie in der Unix-Welt heutzutage üblich, weitestgehend problemlos.

Eine bestehende nagios-Grundkonfiguration konnte von einer alten 2.7-Installation ebenfalls problemlos mit übernommen werden.

Die Anbindung einer mysql-Datenbank an nagios wurde mittels des Eventbrokermoduls und des Daemonprozesses aus den NDOUtils [21] realisiert. Die Konfiguration und Installation dieser beiden Module gestaltete sich ebenso einfach wie die vorangegangene nagios Installation. Mittels spezieller Tools wurde die bestehende Konfiguration der Infrastruktur in die Datenbank übertragen.

Nagios bringt in der Standardinstallation ein schon recht brauchbares Webinterface zur Benutzerinteraktion mit. Zum Zugriff wird ein Webserver benötigt, standardmäßig ist bei vielen Linux-Varianten der Webserver apache installiert. Die Integration von nagios verlief hierbei problemlos.

### **4.6.2 Konfigurationsinterface fruity (kooperierender Ansatz)**

Es existieren mittlerweile eine Vielzahl an frei verfügbaren Tools, um den Konfigurationsvorgang mittels eines Webinterface zu realisieren. Der Vorteil dieser Lösung ist sicherlich in der Tatsache zu suchen, dass die fehleranfällige Konfiguration mittels direktem editieren der Konfigurationsdaten „von Hand“ vermieden wird. Das kostenlose und im Quellcode vorhandene Tool fruity [17] ist von den Möglichkeiten am fortgeschrittensten, kann hier neben der Konfiguration der zu überwachenden Infrastruktur auch der nagios-Systemprozess direkt im Webinterface konfiguriert werden. Abbildung 24 zeigt einen Screenshot der Host-Übersichtsseite des Mailserver irams2.

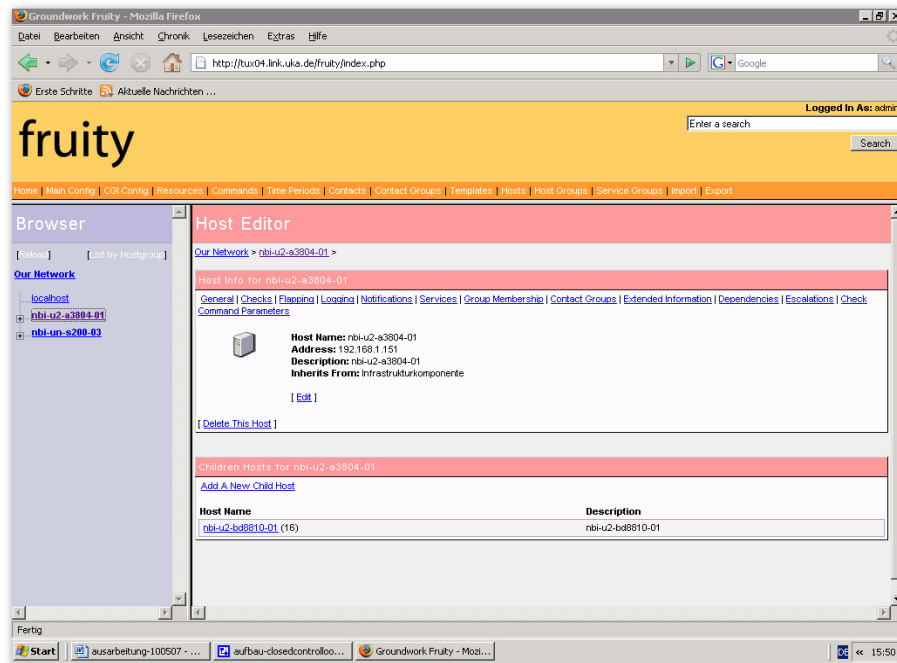


Abbildung 24 Webinterface fruity

### 4.6.3 Groundworks Opensource Monitor (integrierter Ansatz)

Einen auf den ersten Blick interessanten Ansatz scheint die Firma Groundworks mit ihrem in der Einsteigerversion kostenlosen Produkt Groundworks Opensource Monitor [22] (GWOM) zu verfolgen. Nagios wird hier als Teil einer integrierten Monitorlösung verwendet, in dessen Kern der Begriff *Servicemonitoring* steht (siehe **Abbildung 23**). Im Rahmen der Arbeit wurde untersucht, inwiefern dieses Werbeversprechen erfüllt werden kann.

Die Installation des Produktes verlief relativ problemlos, für einen einfachen Test wurde eine standardmäßige Redhat Enterprise Linux 4 durchgeführt. Der Versuch, den GWOM unter FedoraCore 5 zum laufen zu bringen schlug mehrmals fehl. Bei der Installation fällt auf, dass die Software trotz vorhandenem Systemdiensten wie ldap, apache und einer bereits vorhandenen mysql Datenbank zusätzlich diese Komponenten mitinstalliert – bei einer „integrierten Lösung“ ist dies durchaus auch nachzuvollziehen. Es war keinerlei Konfigurationsarbeit notwendig, um das System sofort zu testen. Der Zugang zum GWOM wird durch ein Selbstentwickeltes Webinterface realisiert. Durch dieses Webinterface hat der Benutzer Zugang zu allen Teildiensten des GWOM, insbesondere auch zum integrierten nagios. Es fällt jedoch sofort auf, dass Groundworks hier auf das Webinterface von nagios zurückgreift (siehe voriger Abschnitt), eine eigenständige Entwicklung in diesem Bereich wäre sicherlich wünschenswert gewesen. Das Webinterface des GWOM bringt auch ein Modul zur Modellierung der zu überwachenden Infrastruktur mit. Die Funktionalität ist ausgereift, jedoch keinesfalls überlegen im Vergleich zu den vielen frei verfügbaren Konfigurationsmodulen, die für nagios erhältlich sind. [17][23]

Weiterhin hat sich gezeigt, dass die angepriesenen Funktionen bezüglich Dienstorientierung (siehe auch nächstes Kapitel) nicht erfüllt werden können.

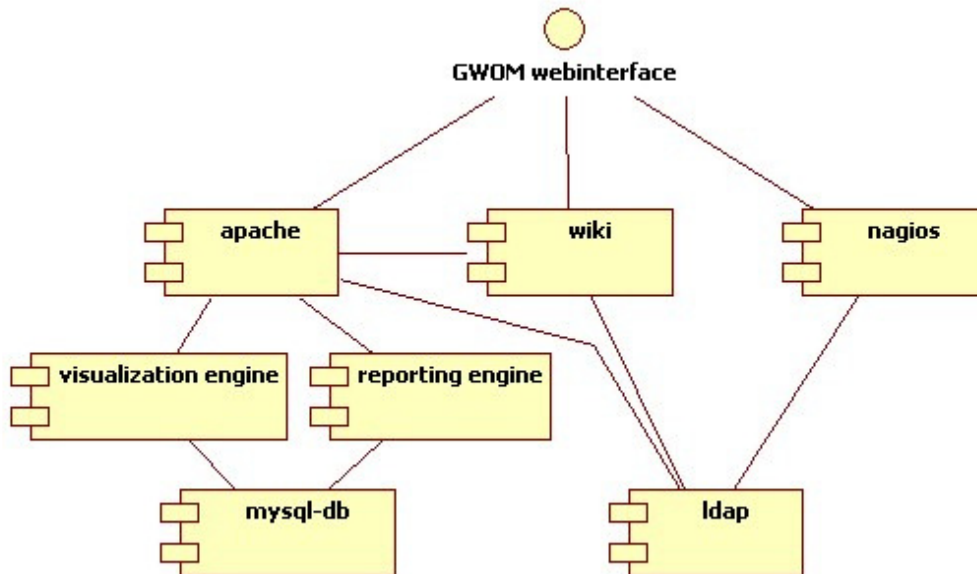


Abbildung 25 Aufbau Groundworks OpenSource Monitor

Trotzdem bleibt der Ansatz von Groundworks interessant, die weitergehende Entwicklung (OpenSource Produkt!) sollte mit Spannung verfolgt werden.

## 4.7 Weiterführende Möglichkeiten

Bedingt durch das Design von nagios kann das Produkt flexibel eingesetzt und erweitert werden. Fehlende Funktionen, vor allem im mitinstallierten Webinterface, können so nachträglich umgesetzt werden.

An erster Stelle wäre es wünschenswert, ein Konfigurationsmodul in die Benutzerschnittstelle zu integrieren. Die Firma Groundworks (siehe voriger Abschnitt) hat gezeigt, wie dies auf einfache Weise geschehen kann. Es bleibt abzuwarten, was sich noch an der Entwicklung der 3er Version von nagios tut, zumal selbige sich noch in einem frühen Stadium befindet. Laut den Angaben der Entwickler auf der Website [24] des Projekts ist für diese Version eine vollständige Neugestaltung der Webschnittstelle von cgi [25] hin zu einer php-basierten [26] Technologie angedacht.

Sind die Möglichkeiten für eine reine Infrastrukturüberwachung in nagios annähernd vollständig und ausreichend implementiert, wäre es wünschenswert, ähnlich der Hoststatus-Map eine Servicestatus-Map zu haben. Während die (physikalischen und/oder logischen) Zusammenhänge der Infrastruktur hauptsächlich für einen Netzwerktechniker von Bedeutung sind, fehlt bisher die Möglichkeit, Zusammenhänge in den Systemprozessen, auch in Bezug auf eine angebotene Dienstleistung, darzustellen.

Dieser Gedanke wird im nächsten Kapitel (Netzwerkmanagement und Dienstorientierung) aufgegriffen und weiter erläutert.

## 4.8 Zusammenfassung

In diesem Kapitel wurde zunächst ausgehend von einem standardisierten Kommunikationsmodell ein System für eine Überwachungsanwendung definiert. Diese Überwachungsanwendung mit nagios als Monitoring-Core wurde schließlich umgesetzt.

Zur Bewertung wurde das in den vorigen Kapiteln eingeführte Mailsystem beispielhaft in die Überwachung aufgenommen. Die Evaluation hat ergeben, dass sich ein flexibles Monitoringsystem mit nagios generell verwirklichen lässt. Der Testbetrieb in der ATIS hat dies auch bestätigt.

Die folgende Gegenüberstellung fasst nochmals die entscheidenden Vorteile von Nagios gegenüber BigBrother auf einen Blick zusammen.

**Tabelle 3 Gegenüberstellung von BigBrother und nagios**

Anforderung	BigBrother	nagios
Datenspeicherung	--	+
Fehlerzustände	o	++
Offene Protokolle	-	+
Manager und Agenten	-	o
Strukturierung	--	++
Abhängigkeiten	--	++
Fehlermanagement	o	++
Heterogener Aufbau	++	++
Erweiterbarkeit	+	+
Abbildung der org. Struktur	--	+
Kosten	++	+

Legende:

- '--' entspricht sehr schlecht
- '-' entspricht schlecht
- 'o' entspricht neutral
- '+' entspricht gut
- '++' entspricht sehr gut

## 4.9 Konkrete Umsetzung der Monitoring-Architektur

Wie die Ergebnisse der bisherigen Kapitel aufzeigen, ist nagios prinzipiell dafür geeignet, im Zuge der Modernisierung der Monitoring-Architektur BigBrother zu ersetzen. Zur Evaluation des Produktes wurde im Testbetrieb auch das mögliche Zusammenspiel mit bestehenden Systemen, etwa Munin[27], getestet und gezeigt, dass nagios problemlos in eine bestehende Infrastruktur integriert werden kann.

Abschließend soll nun in einer konkreten Umsetzung der gewonnen Erkenntnisse gezeigt werden, wie eine zukünftige Überwachungsanwendung in der ATIS aussieht. Die Migrationstrategie sieht vor, zunächst die Infrastruktur vollständig in die Überwachung mit nagios aufzunehmen, sowohl das bestehende System BigBrother und nagios parallel zu betreiben und anschließend nagios als alleiniges System zu verwenden.

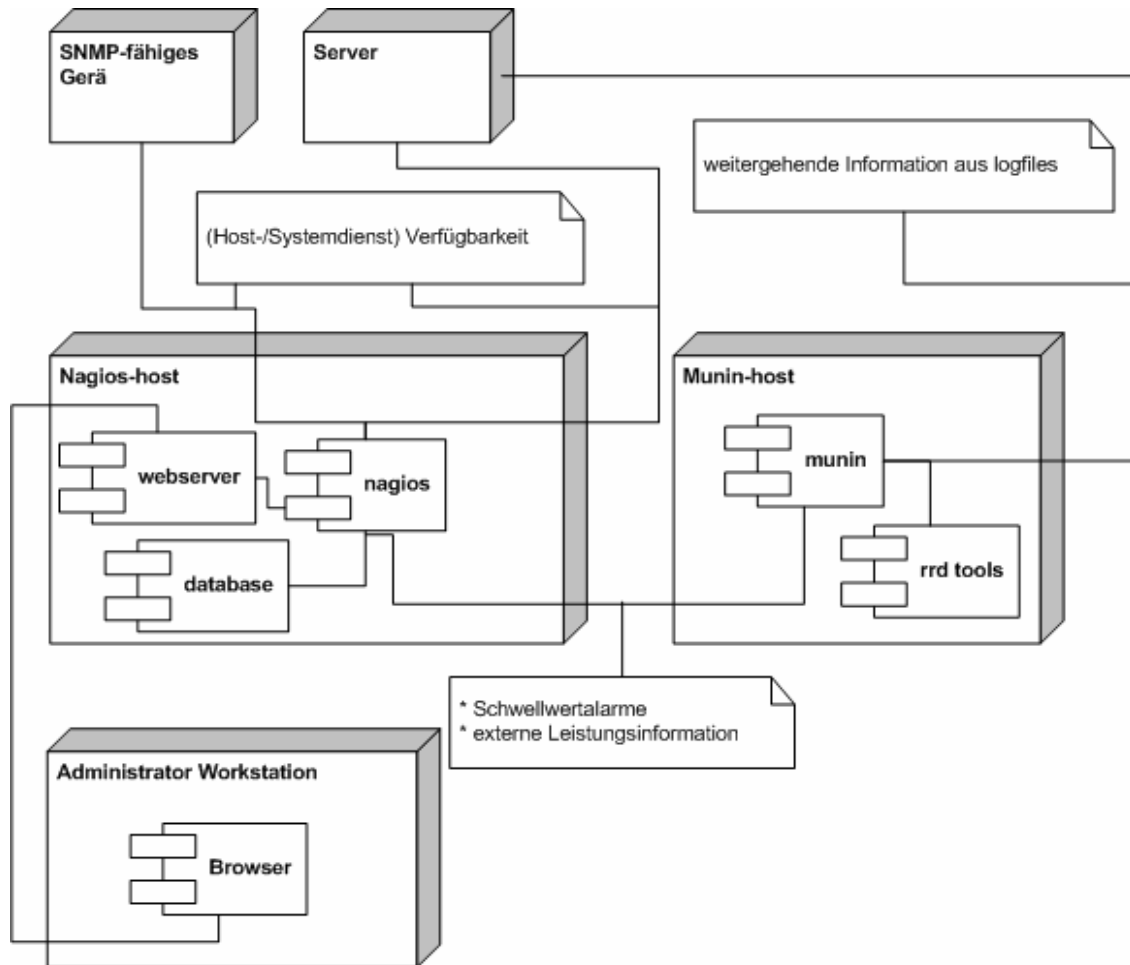


Abbildung 26 konkrete Umsetzung in der ATIS

**Abbildung 26** zeigt die Integration von munin und nagios. Munin wurde entwickelt, um an Information aus log-Files von lokalen Systemprozessen zu gelangen, diese Information einem zentralen Munin-Manager zur Verfügung zu stellen und Leistungsschaubilder zu generieren. Dabei ist die Möglichkeit interessant, Schwellwertangaben zu definieren und bei Überschreiten dieser Schwellwerte eine Alarmmeldung an den nagios-core zu versenden. In der Testumsetzung wurde dieses Setup gewählt, um die Größe der Queuesize des exim Systemprozesses zu überwachen, visualisieren und im Fehlerfall den nagios-core anzustoßen, eine Benachrichtigung zu versenden. Der überwachte Systemprozess ist dabei in nagios integriert und kann innerhalb der Infrastruktur modelliert werden. Dem Munin-Manager kommt somit Proxy-Funktionalität in Bezug auf die Ausführung eines Testplugins zu.

## 5 nagios und Dienstorientierung

Im bisherigen Verlauf der Arbeit wurden grundlegende Definitionen und Begriffe eingeführt, die die Begriffe „Netzwerkmanagement“ und „Dienst“ im Fokus haben. Die Evaluierung im vorigen Kapitel bezog sich hauptsächlich auf den Aspekt, inwiefern nagios als Ersatz für das den heutigen Anforderungen nicht mehr genügende BigBrother im Bezug auf Infrastrukturmanagement dienen kann. In diesem Kapitel gehen wir nun einen Schritt weiter, und untersuchen, inwiefern nagios den Dienst als zu überwachende Komponente in den Vordergrund stellen kann, und wie der Dienst als funktionale Komponente mit einer Infrastruktur zusammenhängt.

### 5.1 Dienstorientierung – der Dienst im Mittelpunkt

In den bisherigen Betrachtungen wurde der Fokus hauptsächlich auf die Überwachung von Hosts, Systemdiensten und deren Abhängigkeiten gelegt. Wie bereits erwähnt, bietet nagios die Möglichkeit, logisch zusammengehörende Systemdienste zu gruppieren und damit eine Dienstinfrastruktur so natürlich wie möglich wieder zu geben.

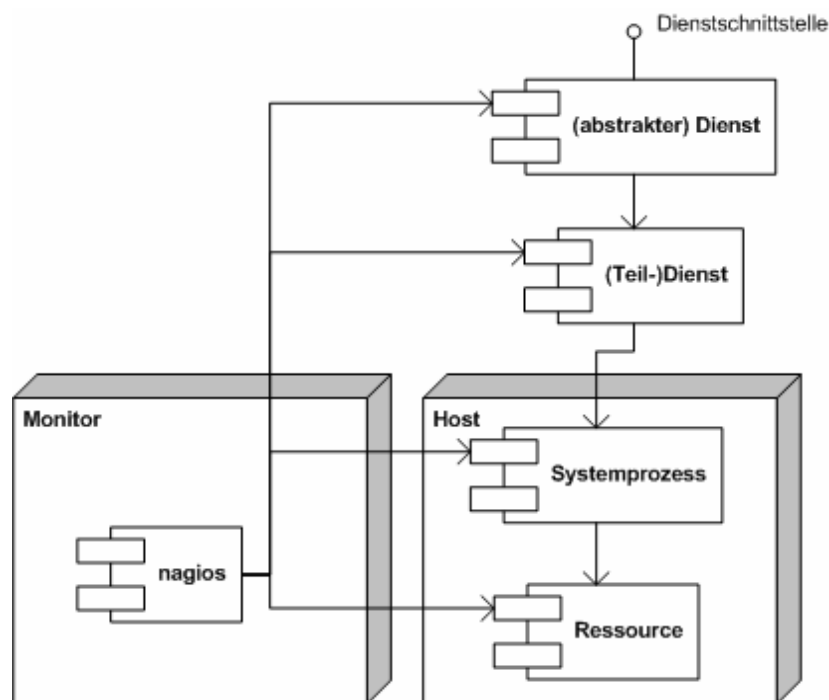


Abbildung 27 Aufbau eines abstrakten Dienstes

Dienstorientierung hat als Konsequenz, dass der Dienst auch im Überwachungsprozess in den Mittelpunkt der Überwachung gestellt wird. Dabei tauchen zwangsläufig zusätzliche Fragestellungen auf:

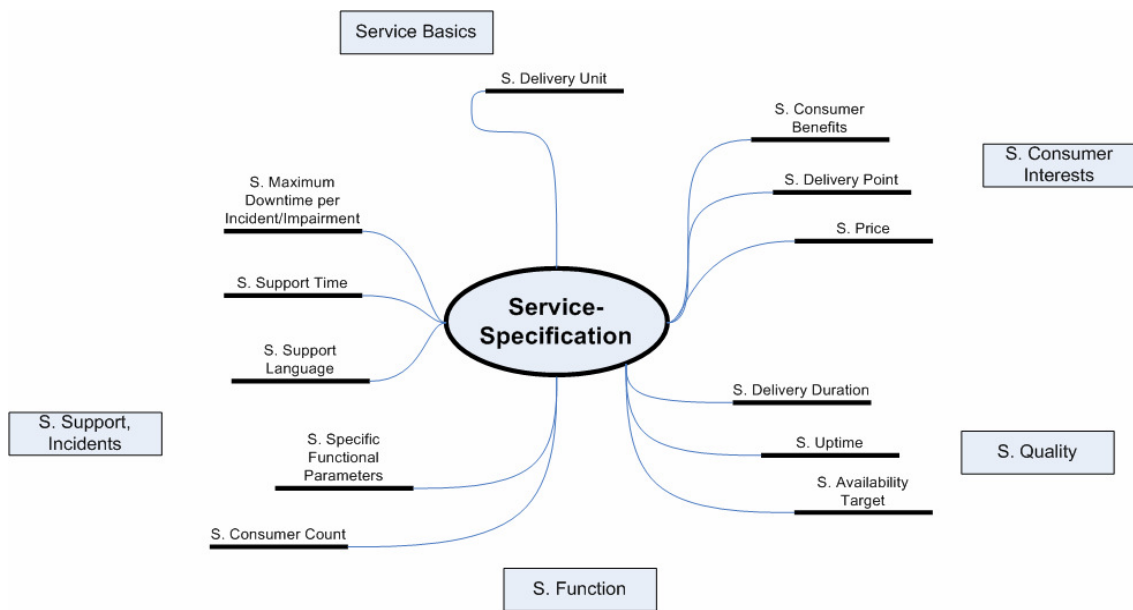
1. Wie kann der Dienst als abstrakte, nicht unmittelbar greifbare Komponente überwacht werden? Im Gegensatz zu einem physikalischen Gerät (Router, Server) kann ein Dienst nicht durch einen Erreichbarkeitsping getestet werden
2. Welche qualitativen Aussagen können getroffen werden? Die Erreichbarkeit eines Routers kann einfach prozentual angegeben werden. Wie würde dies beim





## 5.2 Eigenschaften eines IT-Dienstes

Um die Einhaltung einer festgeschriebenen Dienstleistung überwachen zu können, muss zunächst definiert werden, was diesen Dienst auszeichnet (Dienst-Merkmale). Paul G. Huppertz hat dazu im Rahmen der ITIL-Pocket Guide Reihe in Band 6: IT-Service – Der Kern des Ganzen [31] einige Charakteristika definiert, die einen IT-Dienst vollständig beschreiben zu versuchen. Stellt man diese Eigenschaften in den Fokus der Überwachung, kann der Dienst gesamtheitlich erfasst werden und somit die Überwachung auf diesen Dienst ausgelegt werden. Man kommt damit vom ressourcenbezogenen Monitoring hin zum dienstorientierten Monitoring. In **Abbildung 27** werden zwölf Eigenschaften eines Dienstes beschrieben, wie sie Huppertz definiert.



**Abbildung 29** Eigenschaften eines Dienstes aus Nutzersicht

Versucht man nun, diese Eigenschaften auf den im Laufe dieser Arbeit vorgestellten Mail-Dienst zu übertragen, kommt man zu folgenden Ergebnissen:

Service Delivery Unit	Eine Mail
Service Consumer Benefits	Asynchrone Kommunikation
Service Delivery Point	Email-Programm des Kunden <i>oder</i> webmail-Interface der ATIS
Service Price	Pro Mail <i>oder</i> Pro Zeitraum <i>oder</i> Flatrate
Service Delivery Duration	Maximal 60 Minuten
Service Uptime	24 Stunden, 7 Tage
Service Availability Target	99 %
Service Consumer Count	2000 Nutzer gleichzeitig
Service Specific Functional Parameters	Max. Mailgröße 50 MB, Max. Mailbox-Size 200 MB
Service Support Language	Deutsch/Englisch
Service Support Time	Montag-Freitag 8-16 Uhr
Service Maximum Downtime	8,76h / Jahr

**Tabelle 4** Serviceeigenschaften des Mailteildienstes Mail-Senden

Damit ist der Mailedienst aus Nutzersicht vollständig beschrieben. Mit Hilfe dieser Eigenschaften kann nun die Überwachung der vorhandenen Infrastruktur dahingehend ausgelegt werden, dass die Eigenschaften erfüllt werden.

Schnell fällt auf, dass einige dieser Eigenschaften nicht unmittelbar in den Aufgabebereich eines NMS eingeordnet werden können (siehe hierzu auch Kapitel 2, Anforderungen an ein NMS). In den Überwachungsbereich eines NMS fallen die Punkte *Service Delivery Point (SDP)*, *Service Delivery Duration (SDD)*, *Service Uptime (SU)*, *Service Availability Target (SAT)*, *Service Count (SC)* und *Service Specific Parameters (SSP)*. SDD, SU, SAT und SC können unmittelbar direkt überwacht oder berechnet werden, während bei der Formulierung des SDP bedacht werden muss, dass eventuell Netzwerkinfrastrukturen zu überwachen wären, die außerhalb des Zuständigkeitsbereiches eines IT-Dienstleisters liegen. Aus den funktionalen Eigenschaften (SSP) lassen sich die unmittelbaren Abhängigkeiten zu weiteren IT-Diensten knüpfen. In diesem Beispiel verbirgt sich hinter der Garantie einer Mailbox-Größe von 200 MB die Notwendigkeit, den Plattenspeicherplatz auf den Mailservern zu überwachen.

### 5.3 Anforderungen zur Umsetzung eines Dienstmodells

Im zweiten Kapitel wurde das Mailsystem aus Nutzersicht modelliert und der Bezug zwischen der Dienstleistung Mail mit dessen Teildiensten Mail-Lesen und Mail-Senden und den vorhandenen Ressourcen der Infrastruktur dargestellt (siehe **Abbildung 3**). Dieser Zusammenhang ist vereinfacht in der nachfolgenden **Abbildung 30** dargestellt.

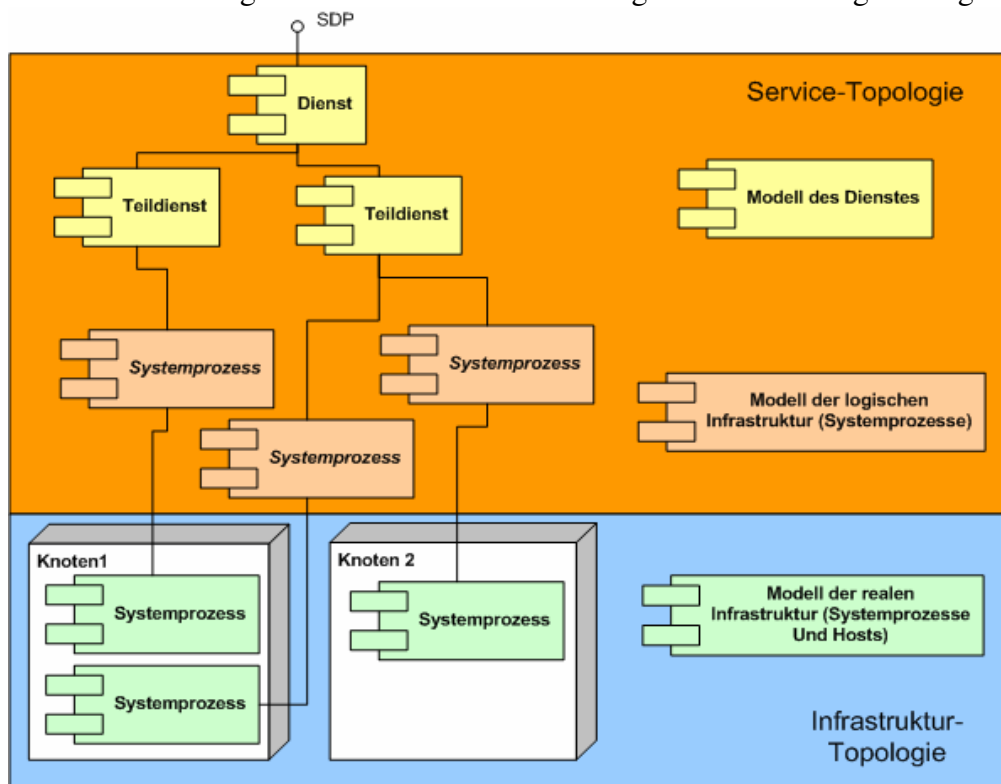


Abbildung 30 Verknüpfung eines Dienstes mit dessen Infrastrukturressourcen

Während für die Überwachung der Komponenten wie Netzwerkgeräte, Serversysteme und Systemprozesse nagios eingesetzt werden kann, stellt sich nun die Frage, inwiefern nagios auch dafür verwendet werden kann, um ausgehend von der Modellierung eines

Dienstes (vergleichbar mit der Modellierung der Infrastruktur) die Verknüpfung der Ressourcen mit der erbrachten Dienstleistung zu überwachen.

Dazu sind folgende Eigenschaften notwendig:

1. *Modellierung des Dienstes*: ähnlich wie die Komponenteninfrastruktur (Hosts, Router, Systemprozesse) müssen IT-Dienste und deren Teildienste modellierbar sein. Dazu müssen spezielle Objekttypen, ähnlich wie Host- und Serviceobjekttypen mit den Eigenschaften aus 5.2, verfügbar sein.
2. *Verknüpfung mit den Ressourcen*: ein IT-Dienst wird von dessen Ressourcen (Systemprozesse, Netzwerke, Hosts) erbracht. Die Überwachung des Dienstes hängt also mit der Frage zusammen, wie die Überwachung der Ressourcen damit verknüpft werden kann. Die prozentuale Verfügbarkeit des Mailedienstes beispielsweise ist eine Aussage über die aggregierte Verfügbarkeit aller seiner einzelnen Komponenten [28].
3. *Visualisierung*: Sowohl die Infrastruktur des Dienstes wie auch die damit verbundene Information muss visualisiert werden. Als Anschauungsbeispiel aus der reinen Infrastrukturüberwachung dient hier die Host-Statusmap (siehe **Abbildung 18**). Dort wird neben dem Aufbau und den Abhängigkeiten der Infrastruktur auch die damit verbundene Verfügbarkeitsinformation (Ergebnisse der Systempings) in Form einer grafischen Webschnittstelle dargestellt. Wünschenswert wäre also insbesondere eine grafische Schnittstelle zur Darstellung des Dienstes (eine Service-Statusmap)

Solch eine „Service-Statusmap“ ist in der aktuellen Version von nagios noch nicht vorgesehen. Durch den offenen und flexiblen Aufbau der Software sollte diese Funktionalität jedoch leicht nachgerüstet werden können. Diese Annahme wird durch die Tatsache gestützt, das sämtliche Leistungs- und Ereignisdaten strukturiert in deiner mysql-Datenbank abgelegt werden.

Es fehlt bisher die Möglichkeit, Services unabhängig von einem Host zu definieren, weshalb die Erfassung der Service-Topologie mithilfe von *Servicegroup* Beschreibungen realisiert wurden. Die Möglichkeiten hierbei sind jedoch eher beschränkt und stark host-bezogen.

Der Screenshot in der folgenden Abbildung zeigt die Sicht auf die Servicegruppen.

The screenshot shows the Nagios web interface in a Mozilla Firefox browser. The main content area displays a 'Service Overview For All Service Groups' with three tables of service status data. The left sidebar contains the Nagios navigation menu.

**Service Overview For All Service Groups**

**Drucken Studpool (Drucken Studpool)**

Host	Status	Services	Actions
i08pr-farbl	UP	2 OK	[Icons]
i08pr-sw1	UP	2 OK	[Icons]
i08pr-sw2	UP	1 WARNING	[Icons]
i08pr-sw3	UP	2 OK	[Icons]

**Geamtheit aller Mail-Teildienste (Mail-Dienst (GESAMT))**

Host	Status	Services	Actions
i08rs2	UP	2 OK	[Icons]
i20smtp	UP	1 OK	[Icons]
i71ms01	UP	2 OK	[Icons]
iram2	UP	1 OK	[Icons]
iramx1	UP	2 OK	[Icons]
iramx2	UP	1 CRITICAL	[Icons]
lists	UP	1 OK	[Icons]
webmail	UP	1 OK	[Icons]

**IMAP/POP Systemprozesse (Mail-Postfachdienst)**

Host	Status	Services	Actions
iram2	UP	2 OK	[Icons]
webmail	UP	1 OK	[Icons]

Abbildung 31 Servicegroup-Sicht im nagios Webinterface

Durch den offenen und modularen Aufbau kann eine Service-bezogene Sicht auf die Infrastruktur leicht realisiert werden. Als Grundlage könnte die bestehende Host-Statusmap herangezogen werden und damit eine Service-Statusmap erstellt werden.

## 6 Zusammenfassung, Ausblick

Die gewonnenen Erkenntnisse dieser Arbeit werden nachfolgend nochmals kurz zusammengetragen, anschließend werden weitergehende Möglichkeiten erläutert, die in dieser Arbeit nicht mehr aufgenommen werden konnten.

### 6.1 Zusammenfassung

Zunächst wurden grundlegende Zusammenhänge erfasst und notwendige Begriffe aus dem Bereich System- und Netzwerkmanagement eingeführt. Begleitend dazu wurde im ersten Kapitel als konkretes Beispiel der von der ATIS angebotene Mail-Kommunikationsdienst beschrieben.

Um verschiedene Produkte miteinander vergleichen zu können, wurden allgemeine Anforderungen an ein System zur Netzwerk- und Systemüberwachung formuliert. Dazu wurde zunächst eine generelle Managementarchitektur beschrieben, die später als Ausprägung in einer Monitorarchitektur umgesetzt werden konnte. Der im ersten Kapitel formal beschriebene Mailedienst wurde modelliert, um so Abhängigkeiten und Gruppierungsmöglichkeiten der beteiligten Systeme und Systemprozesse zu identifizieren.

Anhand der formulierten Anforderungen wurde das bestehende System BigBrother untersucht, dessen Schwachstellen aufgedeckt und mit dem zur Evaluierung vorgeschlagenen nagios verglichen. Die Evaluation im vierten Kapitel wurde von einer Testinstallation begleitet, die sehr gut die Möglichkeiten dieses Produktes gezeigt hat, und schließlich die Entscheidung der ATIS bekräftigt, die Managementinfrastruktur mit nagios als Monitoringkomponente zu modernisieren.

Im letzten Kapitel wurde der Gedanke der Dienstorientierung aus der Einleitung aufgegriffen und auf die aktuellen Fähigkeiten in diese Richtung gehend von nagios gelenkt. Als Wunsch daraus ergibt sich, eine fehlende Service-Statusmap als grafische Schnittstelle zu den in nagios bereits vorhandenen Modellierungsmöglichkeiten zu entwickeln.

### 6.2 Ausblick

Wie die Evaluation und vor allem auch der Testbetrieb gezeigt haben, hat man mit nagios ein äußerst flexibles Tool an der Hand, das sich fast grenzenlos den eigenen Bedürfnissen anpassen lässt. Im Umfeld dieser Arbeit wurden interessante Themengebiete angeschnitten, die bisher teilweise nur kurz (oder gar nicht) Beachtung gefunden haben.

Zunächst bleibt die fehlende Service-Statusmap (siehe voriges Kapitel) zu erwähnen. Sie stellt das fehlende Bindeglied zwischen den in nagios schon vorhandenen Möglichkeiten, Systemprozessabhängigkeiten zu modellieren und in den Kontext einer Dienstleistung zu stellen, dar. Da auf der anderen Seite eine Host-Statusmap aber schon in das Webinterface integriert ist, sollte es mit einwenig Programmiergeschick (Stichpunkt cgi-Tools) möglich sein, diese fehlende Möglichkeit nachzubilden.

Wurde bisher ausschließlich das NMS in Bezug auf die reine Überwachung, also das reine Monitoring der Infrastruktur ausgelegt und der steuernde Eingriff einem Administrator überlassen, kann mithilfe einer flexiblen Überwachungsarchitektur aber auch ein vollautomatisiertes Steuerungsverhalten erzielt werden. Beispielsweise lassen sich Zielvorgaben (so genannte Policies definieren), die beschreiben, welche

Maßnahmen eingeleitet werden müssen, um einen bestimmten Systemzustand (wieder) herzustellen (closed control loop, vergleiche hierzu Abbildung 5 open-control loop).

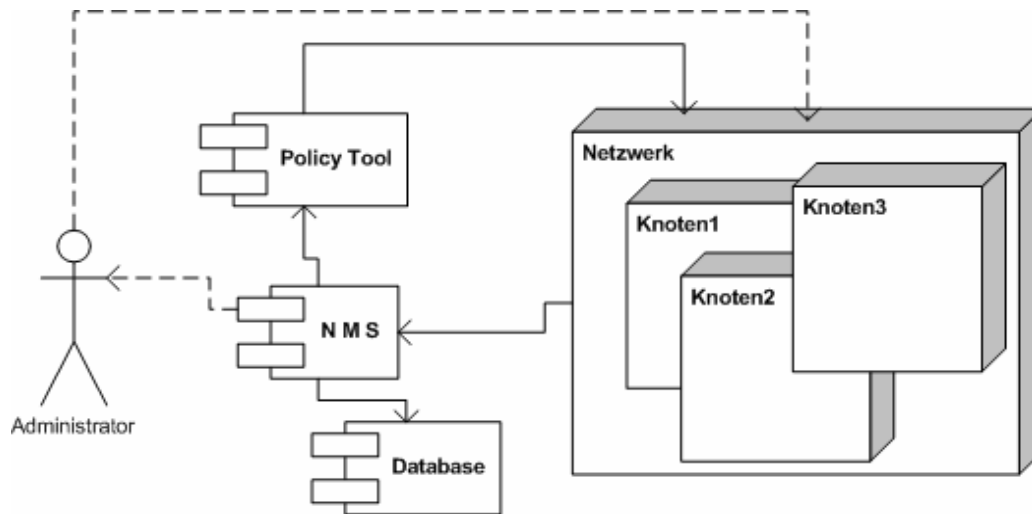


Abbildung 32 closed control loop

Die Diplomarbeit von Thomas Poisl [29] hat zur Zielsetzung, eine Managementarchitektur mit einem automatisierten Policy-basierten Management zu definieren. Mit nagios als Monitoring-Herzstück des Netz- und Systemmanagements kann eine solche Architektur leicht entworfen werden.

Eine weitere Möglichkeit, die Monitoring-Anwendung zu erweitern, besteht im Ausbau der Datenbasis von nagios. Bisher werden die (statischen) Konfigurationsdateien des nagios Prozesses wie auch die der Infrastruktur in einer mysql-Datenbank abgelegt. Die Host-Statusmap greift bei der Visualisierung der Infrastruktur darauf zu und verknüpft diese Daten mit den gemessenen Statusdaten der entsprechenden Komponenten. Die Datenbasis kann noch weiter ausgebaut werden, so dass neben der Konfiguration der Infrastruktur auch beispielsweise die Konfiguration der Service-Topologie, Inventarnummern der vorhandenen Komponenten oder Konfigurationsdaten der Komponenten abgelegt werden. Diese Daten können dann im Rahmen einer Service-Sicht auf die Infrastruktur eingebunden werden.

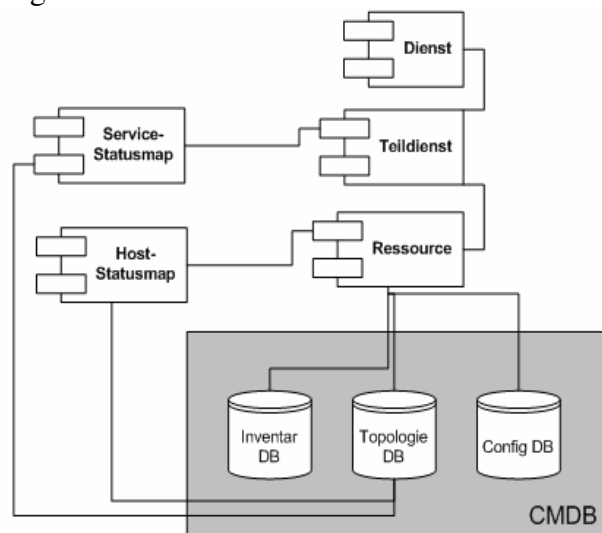


Abbildung 33 Ausbau der Datenbasis

Die bestehende mysql-Datenbank entspricht in diesem Modell der Topologie Datenbank, da hier Informationen über den Aufbau der Infrastruktur gespeichert werden. Zusammengefasst versteht man unter der Speicherung von Topologie, Inventar oder auch Konfigurationsdaten den Begriff der *Configuration Management Database (CMDB)* [32].

## 7 Anhänge

### 7.1 Abbildungsverzeichnisse (Abbildungen und Tabellen)

Abbildung 1 prinzipieller Aufbau eines Dienstes .....	7
Abbildung 2 Vereinfachter Aufbau des Maildienstes .....	10
Abbildung 3 Systematischer Aufbau des Mail-Dienstes.....	13
Abbildung 4 Generelles Modell zur Beschreibung einer Überwachungsarchitektur.....	14
Abbildung 5 open-control loop .....	17
Abbildung 6 Aufbau von BigBrother .....	22
Abbildung 7 Einstiegseite BigBrother .....	26
Abbildung 8 connection Details .....	27
Abbildung 9 Details für den Service connection .....	27
Abbildung 10 Historie für den Service connection.....	28
Abbildung 11 Instanzen einer Monitoring Architektur.....	31
Abbildung 12 Monitoranwendung .....	33
Abbildung 13 Zusammenhänge der Objekttypen - Klassendiagramm .....	34
Abbildung 14 Zusammenhang zwischen Objekten und nagios Modellierung .....	35
Abbildung 15 Integration NDO ( <u>N</u> agios <u>D</u> ata <u>O</u> bjects).....	37
Abbildung 16 Integration RRD-Tools.....	38
Abbildung 17 Einstiegsseite - Taktische Übersicht .....	38
Abbildung 18 Hoststatus Übersichtsseite.....	39
Abbildung 19 Servicestatus Übersichtsseite .....	39
Abbildung 20 Servicegruppen Übersichtsseite .....	40
Abbildung 21 Visualisierung der Infrastruktur .....	40
Abbildung 22 Eingebettete Leistungsinformation .....	41
Abbildung 23 Vergleich: isolierter, kooperierender, integrierter Ansatz.....	41
Abbildung 24 Webinterface fruity .....	43
Abbildung 25 Aufbau Groundworks Opensource Monitor.....	44
Abbildung 26 konkrete Umsetzung in der ATIS .....	46
Abbildung 27 Aufbau eines abstrakten Dienstes .....	47
Abbildung 28 Zusammenhang der Metriken unterschiedlicher Schichten .....	48
Abbildung 29 Eigenschaften eines Dienstes aus Nutzersicht .....	49
Abbildung 30 Verknüpfung eines Dienstes mit dessen Infrastruktururessourcen .....	50
Abbildung 31 Servicegroup-Sicht im nagios Webinterface.....	52
Abbildung 32 closed control loop .....	54
Abbildung 33 Ausbau der Datenbasis.....	54
Tabelle 1 Aufbau und Organisation der Arbeit.....	12
Tabelle 2 Zusammenfassung der Anforderungen an ein NMS .....	20
Tabelle 3 Gegenüberstellung von BigBrother und nagios .....	45
Tabelle 4 Serviceeigenschaften des Mailteildienstes Mail-Senden .....	49



## 7.2 Index

Abhängigkeiten .....	13, 25, 30, 47	Hauptaufgabe eines NMS .....	18
impliziete, expliziete .....	16	Heterogenitätsbedingung .....	18
Agent.....	14, 15, 21, 22	Host .....	34
Agenten.....	23, 26, 31	Hostdependancy .....	34
agentenbasierter Test .....	23	Hostescalation .....	35
agentenloser Test .....	24	Hostgroup.....	34
Alarmbericht .....	23	Hoststatusmap .....	51
Anforderungen .....	53	IBM Tivoli .....	29
apache .....	42	IMAP.....	9
ATIS.....	9, 53	Information Model .....	14
Basisdienst .....	10	Informationsmodel .....	26
BigBrother .....	12, 21, 33	Infrastrukturkomponente.....	30
Bottom-Up Ansatz .....	48	Infrastrukturmanagement.....	47
CIM.....	14, 26, 37	Institutsmailserver .....	9
Client.....	21	IT-Dienst .....	48
closed control loop.....	54	IT-Dienste .....	7
Command.....	34	ITIL .....	49
Communication Model .....	14	IT-Infrastruktur .....	7
Contact .....	34	IT-Management.....	7
Contactgroup.....	34	IT-Systeme .....	7
Controlling.....	8, 21, 29	Kommunikationskanäle .....	19
Courier .....	9	LDAP .....	9
Data Storage.....	31	Leistungsdaten .....	31
Database.....	14	Leistungsmessdaten .....	36
Data-Warehouse Systeme .....	14	LINK .....	9
Datenmodell.....	37	Maildienst .....	13
Datenspeicherung.....	14, 36	Mail-Exchangern.....	9
Dienst.....	7, 22, 47	Mailtransfer-Agent.....	9
Dienstanbieter .....	7	MTA.....	9
Dienstgüte .....	17	Managed Node .....	15
Dienstleistungsvereinbarungen .....	7	Managed Nodes .....	8, 21
Dienstnehmer .....	7	Managed Objects .....	15, 22
Dienstorientierung .....	12	Management Interface.....	31
Dienstorientierung .....	47	Managementarchitektur .....	16, 53
Email-Dienst .....	9	Management-Architektur .....	14
Erreichbarkeitsping .....	47	Manager .....	14, 15, 21, 22
Event-Broker Module .....	37	Monitorarchitektur .....	53
Exim.....	9	Monitoring .....	8, 21, 28, 31
FCAPS .....	8, 17	Monitoring Architektur.....	31
Fehl- bzw. Folgefehleralarme .....	19	Monitoring-Architektur.....	11
Fehlerzustände .....	15	Monitoring-Core .....	31
Framework Charakter .....	32	Munin .....	45
fruity.....	32, 41	mysql.....	36, 51
Function Model.....	14	nagios .....	12, 31, 47
Groundworks Opensource Monitor ...	41	NDOUtils .....	42
grundlegende Anforderungen .....	13	Netzinfrastruktur .....	17
Gruppierung		Netzwerkping .....	24
impliziete, expliziete .....	15	NMS .....	8

29	
NRPE.....	32
NSCA .....	32
objektorientierte Funktionen .....	30
open-control loop .....	17
opendap .....	10
openNMS .....	29
organisatorische Struktur.....	19
Organization Model.....	14
Performance Data Visualisation.....	31
Plugins .....	21, 32
Plug-Ins .....	19
policy-based Management.....	12
POP.....	9
Protokoll.....	16
Quality of Service.....	36
Ressourcen .....	14
RRDTools.....	33
Schwachstellenanalyse .....	21
Server .....	21
Service.....	34
Service dependancy .....	34
Servicegroup.....	34
Servicescalation.....	35
Servicestatusmap .....	51
SMTP.....	9
SNMP .....	16, 30, 32
Systemdienste .....	36
Systemdienstes .....	8
Teildienste .....	13, 36
Templates .....	30
Testinstallation .....	42
Timeperiod .....	34
Top-Down Ansatz .....	48
Überwachung.....	14
Überwachungssystem .....	14
Userinterface .....	14
Visualisierung.....	37, 51
Webinterface .....	9, 26, 28, 38
Weboberfläche.....	24, 29
Weitere Kriterien .....	20
Ziele .....	11

### 7.3 Abkürzungen und Glossar

<b>Abkürzung oder Begriff</b>	<b>Bezeichnung oder Beschreibung</b>
Agent	Stellvertreter auf einer zu verwaltenden Einheit, die Information über diese Komponente zur Verfügung stellt und den Zugriff auf die Attribute der Komponente ermöglicht
ATIS	Bezeichnung für eine Abteilung innerhalb der Fakultät für Informatik an der Universität Karlsruhe. Die <i>Abteilung Technische Infrastruktur</i> hat zur Aufgabe, eine Netzwerkinfrastruktur und darauf aufbauend, bestimmte Systembasierte Dienste, für die übrigen Institute der Fakultät bereitzustellen
Basisdienst	Systemdienst, der einem Dienstanutzer nicht unmittelbar zur Verfügung steht. Ein <i>Basisdienst</i> übernimmt grundlegende Aufgaben und wird zumeist mit weiteren Diensten komponiert um daraus einen einem Nutzer zugänglichen Dienst zu bilden.
BigBrother	Softwaresystem zur Überwachung von (physikalischen) Infrastrukturkomponenten und in gewissem Maße auch für Systemdienste
Bottom-Up Ansatz	Bei der Umsetzung des Modells des Mailsystems wurde zunächst von den einzelnen Komponenten der Infrastruktur ausgegangen, um diese anschließend in den Kontext eines Dienstes zu stellen
CIM	Das <i>Common Information Model</i> beschreibt Meta-Modelle, mit deren Hilfe Information beschrieben, abgelegt und in Relation zueinander gestellt werden kann
closed-control loop	Mechanismus, bei dem kein menschliches Eingreifen zur Steuerung notwendig ist
Communication Model	Beschreibt Beziehungen zur Kommunikation verschiedener Komponenten innerhalb der Management-Architektur
Controlling	Steuernder Eingriff auf Komponenten der Infrastruktur
Dienst	Zusammenfassung der technischen Parameter eines Systems hinter einer funktionalen Schnittstelle. Die Produktion eines immateriellen Dienstes entspricht der Produktion einer materiellen Ware in der realen Welt
Dienstanbieter	Anbieter eines Dienstes. Meist wird dies mit einer Firma,

	Einheit o.ä. gleichgesetzt, kann aber auch eine Softwarekomponente bezeichnen
Dienstnehmer	Nutzer eines Dienstes. Kann eine reale Person aber auch eine Softwarekomponente bezeichnen
Dienstleistungsvereinbarung	Vertragliche Vereinbarung zwischen einem Dienstnehmer und einem Dienstanbieter über die Erbringung einer Dienstleistung.
Dienstgüte	Bezeichnet die Qualität der Erbringung eines Dienstes
Dienstorientierung	Im Kontext des Monitorings bedeutet <i>Dienstorientierung</i> , das der Dienst, der von einer Infrastruktur erbracht wird, in den Vordergrund der Überwachung gestellt wird, und das die überwachten Ressourcen dieser Infrastruktur den funktionalen Eigenschaften eines Dienstes zugeordnet werden
Email-Dienst	Asynchroner Kommunikationsdienst. Der <i>Email-Dienst</i> ist einer der zentralen Dienste, die von der ATIS angeboten werden
FCAPS	Abkürzung für die englischen Begriffe <i>Fault-, Configuration-, Accounting-, Performance-, Securitymanagement.</i>
Function Model	Beschreibt funktionale Anforderungen eines Systems innerhalb der Management Architektur
Host	Materielle Komponente einer Infrastruktur. Ein <i>Host</i> ist die Maschine, auf der Systemdienste in Form von Systemprozessen ausgeführt werden
IMAP	Das <i>Internet Mail Access Protocol</i> beschreibt ein Protokoll zum Zugriff auf Mailboxen über das Internet.
IT-Dienst	Dienstleistung, die von Komponenten einer IT-Infrastruktur erbracht werden
ITIL	Die <i>Information Technology Infrastructure Library</i> bezeichnet eine Sammlung von Dokumenten des englischen Wirtschaftsministeriums mit der Zielsetzung, <i>Best Practice</i> Ansätze zur Lösung von wiederkehrenden Problemen bezüglich IT zu formulieren.
IT-Infrastruktur	Gesamtheit der Hosts, Netzwerkkomponenten und Systemdienste
IT-Management	Überbegriff

---

IT-System	Bezeichnet ein System innerhalb der IT-Infrastruktur, das einen oder mehrere IT-Dienste erbringt. Kann sowohl eine einzelne Komponente wie beispielsweise einen Router bezeichnen, aber auch eine Sammlung von Komponenten
LDAP	Das <i>Leightweight Directory Access Protocol</i> beschreibt ein Protokoll zur Abfrage von Verzeichnisdatenbanken
Managed Node	Repräsentation einer zu überwachenden Komponente
Managed Object	Eine oder mehrere zu managende Eigenschaft einer Komponente werden durch ein Managed Object beschrieben.
Management-Architektur	Beschreibung einer Architektur zum Überwachen <i>und</i> Steuern einer Infrastruktur
Manager	Komponente, die auf einen Agenten einwirkt, um entweder Information zu erhalten, diese aufzubereiten und einem Nutzer zugänglich macht, oder aber steuernd auf einen Agenten einwirkt
Monitoring	Vorgang des Überwachens
Monitoring-Architektur	Teil der Management-Architektur, die die Möglichkeiten zum direkten steuernden Eingriff vernachlässigt
Nagios	Opensource Produkt zur Überwachung von Hosts und Systemdiensten. Modular aufgebaut, so das es leicht erweitert werden kann
open-control loop	Erfolgt der steuernde Eingriff auf eine Infrastrukturkomponente durch die Handlung eines Administrators, spricht man von einem open-control loop
Organisatorische Struktur	Beschreibt den Aufbau und die Hierarchiebeziehungen einer Organisation
Organization Model	Modellbeziehung innerhalb der Managementarchitektur, die den Aufbau eines Managementsystem beschreibt
Service	Englisches Wort für Dienst, wird im Deutschen als Synonym verwendet
SMTP	Das <i>Simple Mail Transfer Protocol</i> beschreibt, wie Emails in Netzwerken transportiert werden.
SNMP	Das <i>Simple Network Management Protocol</i> beschreibt

	Verfahren zum Zugriff zum Überwachen und Steuern von Netzwerkkomponenten.
Systemdienst	Dienstleistung, die unmittelbar von einem Systemprozess erbracht wird. Meist wird Systemdienst und Systemprozess als Synonym verwendet.
Top-down Ansatz	Bei der Frage nach Dienstorientierung wird von einer zu überwachenden Dienstleistung ausgegangen, um diese nach und nach zu Verfeinern um schließlich den Fokus auf die Komponenten der Infrastruktur zu bringen.
Webinterface	Benutzerschnittstelle, die mittels eines üblichen Webbrowsers genutzt werden kann. Voraussetzung ist die Integration der Programmlogik in die Funktionalität eines Webservers

## 7.4 Literatur und Quellenverweise

- [1] Hegering, Abeck, Neumaier; 1999  
*Integriertes Management verteilter Systeme*
- [2] Physikalischer Aufbau des Datennetzes  
[http://www.atis.uka.de/img/content/LINK\\_physikalisch\\_web\\_big.gif](http://www.atis.uka.de/img/content/LINK_physikalisch_web_big.gif)
- [3] SMTP - Simple Mail Transfer Protocol  
<http://www.ietf.org/rfc/rfc2821.txt>
- [4] <http://www.exim.org/>
- [5] IMAP – Internet Message Access Protocol  
<http://www.ietf.org/rfc/rfc2060.txt>
- [6] POP – Post Office Protocol <http://www.ietf.org/rfc/rfc1939.txt>
- [7] Horde webmail Interface  
<http://webmail.ira.uka.de>
- [8] Implementierung eines freien ldap Servers  
<http://www.openldap.org/>
- [9] Christian Mayerl; 2006  
*Vorlesung SESM – Service Engineering, Service Management*  
<https://bscw.ira.uni-karlsruhe.de/pub/bscw.cgi/774632>
- [10] SNMP – Simple Network Management Protocol  
<http://www.ietf.org/rfc/rfc1157.txt>
- [11] CIM – Common Information Model  
<http://www.dmtf.org/standards/cim/>
- [12] Thorsten Tüllmann; Studienarbeit 2005  
*Rechner- und Dienstüberwachung in den heterogenen Netzen des Rechenzentrums*  
<http://dsn.tm.uni-karlsruhe.de/922.php>
- [13] IBM Tivoli Management Suite  
[http://de.wikipedia.org/wiki/Tivoli\\_IBM](http://de.wikipedia.org/wiki/Tivoli_IBM)
- [14] <http://www.opennms.org>
- [15] Network and Hagios  
<http://www.nagios.org>
- [16] <http://www.nagios.org/about/>
- [17] Groundwork fruity  
<http://fruity.sourceforge.net/>
- [18] Tobias Oetiker  
<http://oss.oetiker.ch/rrdtool/>
- [19] [http://nagios.sourceforge.net/docs/3\\_0/objectdefinitions.html](http://nagios.sourceforge.net/docs/3_0/objectdefinitions.html)
- [20] [http://nagios.sourceforge.net/docs/ndoutils/NDOUtils\\_DB\\_Model.pdf](http://nagios.sourceforge.net/docs/ndoutils/NDOUtils_DB_Model.pdf)
- [21] <http://www.nagios.org/download/>
- [22] <http://www.groundworkopensource.com/products/os-overview.html>
- [23] PHP-based Management Interface  
<http://www.nagiosql.org/>
- [24] <http://www.nagios.org/development/upcoming.php>
- [25] [http://de.wikipedia.org/wiki/Common\\_Gateway\\_Interface](http://de.wikipedia.org/wiki/Common_Gateway_Interface)
- [26] <http://www.php.net/>
- [27] <http://munin.projects.linpro.no/>
- [28] Mayerl, Hüner, Gaspar, Momm, Abeck; 2007  
*Definition of Metric Dependencies for Monitoring the Impact of Quality of*

- [29] *Services on Quality of Processes*  
Thomas Poisl; Diplomarbeit 2006  
*Policy-basiertes Management einer IT-Infrastruktur am Beispiel des VPN-Dienstes*  
<https://bscw.ira.uni-karlsruhe.de/bscw/bscw.cgi/151668>
- [30] Quest Software, BigBrother  
<http://www.bb4.com/>
- [31] Paul G. Huppertz; 2007  
*ITIL-Pocket Guide - Band 6: IT-Service – Der Kern des Ganzen*
- [32] [http://de.wikipedia.org/wiki/Configuration\\_Management\\_Database](http://de.wikipedia.org/wiki/Configuration_Management_Database)
-