Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft

DHBW
Duale Hochschule
Baden-Württemberg
Karlsruhe

Entwicklung eines Nagios Plugins zur Überwachung und Auswertung von Funktionen und Fehlern in Content-Managment-Systemen

BACHELORARBEIT

für die Prüfung zum
Bachelor of Engineering

des Studienganges

Informationstechnik

an der Dualen Hochschule Karlsruhe

von

Andreas Paul

Bearbeitungszeitraum:	25.05.2009 – 23.08.2009
Matrikelnummer:	108467
Kurs:	TIT06GR
Praxissemester:	6
Ausbildungsfirma:	Forschungszentrum Karlsruhe GmbH (FZK) Steinbuch Centre for Computing Hermann-von-Helmholtz-Platz 1 76344 Eggenstein-Leopoldshafen
Betrieblicher Betreuer:	Dr. Doris Wochele
Prüfer der DHBW Karlsruhe:	Holger Raff

Inhalt

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides Statt, dass ich die vorliegende Arbeit selbst angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.
Die Arbeit wurde bisher keiner Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.
Ich versichere hiermit wahrheitsgemäß, die Arbeit bis auf die dem Aufgabensteller bereits bekannte Hilfe selbständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderung entnommen wurde.

Karlsruhe, den 29. Juli 2009

.....

Ort, Datum (Andreas Paul)

1 Einleitung

Einleitung halt. Kurz was ist Nagios, warum überhaupt überwachen? Was soll überwacht werden -> Stellent/UCM kurz was ist das? Warum gerade das überwachen -> Aktive Benutzung durch User - kritisch

2 Abstract

Zusammenfassung von allem.

Aufgabenstellung, Erwartendes Ergebnis

3 Aufgabenstellung

Oracle UCM werden im FZK eingesetzt, bisher nur rudimentäre Überwachung durch Nagios möglich. Diese Arbeit soll die spezifischen Überwachungselemente erurieren und umsetzten.

4 Grundlagen

In diesem Kapitel werden die grundlegenden Begriffe erläutert, die für das Verständnis der weiterführenden Kapitel notwendig sind.

4.1 Überwachungssysteme

was ist wichtig was nicht, gewichtung, klassifizierung, organisationsstrategie

HOst,Services erklären

Geräte nicht nur Server bzw. Rechner, sondern auch Switches Router, oder auch explizite Hardwarekomponenten wie Sensoren für Temperatur, Luftfeuchtigkeit oder Rauchmelder.

4.1.1 Ressourcenbelastung

Die Einführung einer Überwachungssoftware bringt bei größeren Serverlandschaften eine nicht zu verachtende Netzwerk- und Prozessorbelastung mit sich. Dabei gilt es die anfallende Belastung durch zwei unterschiedliche Arten der Überwachung zu unterscheiden:

Lokale / Zentrale Bearbeitung Die Durchführung der Überprüfungen findet durch einen zentralen Überwachungsserver statt, der die Informationen über die einzelnen Hosts und Services über das Netzwerk abfrägt / abfragt. Diese Methode ist in der Regel vorzuziehen, da hierbei die zu überwachenden Geräte weniger belastet werden und die Konfiguration der einzelnen Kontrollschritte zentral möglich / realisierbar ist.

Entfernte / Ausgelagerte Bearbeitung Bei einer sehr hohen Anzahl von zu überwachenden Objekten ist eine zentralisierte Ausführung nicht mehr von einem einzelnen Server tragbar. In diesem Fall ist das Überwachungssystem darauf angewiesen, dass die einzelnen Hosts die kontrollierenden Überprüfungen selbständig durchführen und deren Ergebnisse an den Überwachungsserver weiterzuleiten.

Nagios bietet zusätzlich noch eine weitere, dritte Möglichkeit durch das *Distributed Monitoring* (Verteilte Überwachung) an, siehe Kapitel 5.4.

4.1.2 Netzwerkstruktur und Abhängigkeiten

Die Überwachung von Hosts und Services über das Netzwerk erzeugt normalerweise immer zusätzlichen IP-Traffic. Das bedeutet, dass jede Überquerung weiterer Netzwerkknoten, die zwischen dem Überwachungsserver und den zu überwachenden Geräten liegen, eine weitere Belastung für das Netzwerk bedeutet, sowie eine Abhängigkeit zwischen Host und Server einführt.

Quelle: [Jose07] S. 5

Abbildung 1: Zusätzliche Netzwerkabhängigkeit und Netzwerkbelastung[1]

In der obigen Abbildung erzeugt der Router 1 die zuvor beschriebene zusätzliche Netzwerkabhängigkeit und Netzwerkbelastung, da der Server 1 bei einem Ausfall des Routers nicht mehr durch den Überwachungsserver erreichbar ist und jede Überprüfung, die vom Überwachungsserver gesendet wird den Router mit dem Routing der Pakete belastet.
Deshalb gilt es laut [Jose07] S. 5 folgende zwei Punkte beim Erstellen eines Überwachungssystems zu beachten:

Überwachungsredundanzen vermeiden Redundate Überwachung entsteht dadurch, dass der gleiche Service durch zwei Arten mit unterschiedlichen Tiefen / Tiefgang geprüft wird. Ein einfaches Beispiel ist die Überwachung eines Webservers auf dem Standardport 80. Eine Überwachungsmethode ist es diesen Port abzufragen und die entsprechende Rückantwort des Servers auszuwerten. Soll die auf dem Webserver laufende Webseite überwacht werden, kann die jeweilige Webseite über die Adresse nach einem bestimmten Inhalt untersucht werden.
In beiden Fällen wird getestet, ob der Webserver über das Netzwerk ansprechbar ist, jedoch sagt der zweite Test zusätzlich noch aus, dass die Webseite

[1] Quelle: [Jose07] S. 5

korrekt angezeigt wird, somit wäre der erste Test überflüssig. Jedoch muss zuvor abgewogen werden, ob eine redundante Überwachung nicht sogar hilfreich bei der Ermittlung der Fehlerursache ist. Wenn im oberen Beispiel der Inhalt der überwachten Webseite verändert wird, ist dies nur aus dem zweiten Test ersichtlich.

Minimale Netzwerkbelastung Um bereits stark belastete Netzwerkpunkte zu entlasten, bietet es sich an die Frequenz mit der die Test über das Netzwerk gesendet werden zu verringern. Die Aufstellung des Überwachungsservers ist dadurch gerade bei größeren Serverlandschaften sehr wichtig, da durch eine effiziente Platzierung womögliche Flaschenhälse / Engstellen in Form von veralteten Switches oder ähnlichem vermieden werden können.

4.1.3 Sicherheitsaspekte

Um erweiterte Statusinformationen über einen Prozess oder über die Arbeitsspeicherauslastung auszulesen ist (meistens) zusätzliche Software auf den Hosts nötig. Diese Software benötigt einen zusätzlichen geöffneten Port auf dem zu überwachendem Rechner, die einen neuen Angriffspunkt für Angreifer darstellen kann. Außerdem erhält der Überwachungsserver Ausführungsrechte auf dem Client, so dass eine weitere potentielle Sicherheitslücke in einem (vermeindlich) zuvor sicherem System entsteht. Jeder, der die Kontrolle über den Überwachungsserver besitzt oder sich als solcher ausgibt, kontrolliert somit gleichzeitig alle anderen überwachten Hosts.
Um dies zu verhindern gibt es verschiedene Ansätze. Als ersten Ansatz sollte der Port durch den der Überwachungsserver mit dem Host kommuniziert vom Standardwert abweichen, damit nicht sofort erkennbar ist, dass sich eine (womöglich) ausnutzbare Überwachungssoftware sich auf dem Rechner befindet. Damit die über diesen Port versendeten Informationen nicht für Dritte zugänglich sind, bietet es sich an die auszutauschende Informationen mit einem Algorithmus zu verschlüsseln. Durch den Einsatz eines Verschlüsselungsalgorithmus werden die Informationen nicht mehr im Klartext ausge-

tauscht, sondern Da die Möglichkeit einer Verschlüsselung der Datenübertragung nicht von jeder Überwachungssoftware angeboten wird, gilt diese Option als Auswahlkriterium im produktivem Betrieb. (Verweiss auf Windows Agenten Übersicht?)
Die Erlaubnis der Abfrage der Überwachungsinformationen sollte anhand der IP-Adresse beschränkt werden, so dass der Client nur Anfragen des Überwachungsservers akzeptiert. Durch diese Einschränkung kann vermieden werden, dass sensible Informationen aus den Antworten an unberechtigte Dritte übermittelt werden oder ein Denial of Service-Angriff (DoS) durch eine übermäßig hohe Anzahl an Anfragen an den Client gesendet wird, um eine Überlastung des Servers zu erreichen und diesen somit arbeitsunfähig zu machen.

4.2 Dokumenten-Management-Systeme

Um ein Dokumenten-Management-System (DMS) zu erläutern muss sich zuerst mit dem Begriff des **„Dokuments“** auseinander gesetzt werden. In [DMS08] S. 2 wird ein Dokument durch folgende Punkte definiert:

- Ein Dokument fasst inhaltlich zusammengehörende Informationen strukturiert zusammen, die nicht ohne erheblichen Bedeutungsverlust weiter unterteilt werden können.
- Die Gesamtheit der Information ist für einen gewissen Zeitraum zu erhalten.
- Dokumente dienen oft dem Nachweis von Tatsachen.
- Ein Dokument ist als Einheit ablegbar (speicherbar) und/oder versendbar und/oder wahrnehmbar (sehen, hören, fühlen).
- Das Dokument ist eigentlich der Träger, der die Informationen speichert, egal ob das Dokument ein Stück Paper, eine Datei auf einem Rechner, ein Videoband oder eine Tontafel etc. ist. Dies bedeutet auch, dass es keine Bindung an Papier oder ein geschriebenes Wort gibt.

Desweiteren gibt es eine Differenzierung in zwei Definitionen:

> „Als **Dokument im konventionellen Sinne** werden Dokumente bezeichnet, die als körperliches Dokumente (z. B. Papier) vorliegen, ursprünglich als körperliches Dokument vorlagen oder für die Publizierung auf einem körperlichen Medium vorgesehen sind.
>
> Die Begrifflichkeit des **Dokuments im weiteren Sinne** erweitert den Begriff des Dokuments um semantisch zusammengehörende Informationsbestände, die für die Publikation in nicht-körperlichen Medien, z. B. Webseiten, Radio, Fernsehen o. ä. vorgesehen sind. Derartige Dokumente werden oft dynamisch gestaltet und zusammengestellt."
>
> [DMS08] S. 2

Dabei müssen auch Daten und Dokumente voneinander abgegrenzt werden. In [DMS08] S. 33 werden Daten im Allgemeinen als eher stark strukturierte Informationen gesehen, wobei Dokumente zumeist aus unstrukturierte bis zu schwach strukturierte Informationen bestehen. Eine eindeutige Klassifizierung eines vorhandenen Dokumentes lässt ist jedoch nicht immer möglich, da sich oft Mischungen beider Klassen finden (lassen). Ohne die dazugehörigen Metadaten besteht ein (graphisches) Bild aus unstrukturierten Informationen, daher auch NCI-Dokument für None-Coded Information genannt. Der Anteil von strukturierten Informationen in einem Dokument nimmt von Bildern über Text zu Tabellen zu, da hier die Dokumente vollautomatisch auswertbar sind, siehe hierzu Abbildung ???.

Bild [DMS08] S. 33

Die Einordnung, wann ein Dokument strukturierte oder unstrukturierte Informationen enthält, lässt an folgenden Beispielen verdeutlichen. Bei einem Bild oder Foto lassen sich die enthaltenen Informationen nicht durch Computer bestimmen. Beispielsweise, ob sich eine Person auf dem Bild befindet oder

wann und wo das Foto erstellt wurde. Daher ist ein Bild, solange keine Metadaten darüber bekannt sind, ein eindeutiges Beispiel für NCI-Dokumente mit unstrukturierten Informationen. Im Gegensatz dazu lassen sich die Werte einer Tabelle oder eines Datensatzes durch die Spaltennamen eindeutig bestimmen und durch den Computer auslesen. Solche Daten mit strukturierten Informationen werden daher auch als Dokumententyp mit Coded Information (CI) bezeichnet.
Unter **Dokumenten-Management** werden primär die Verwaltungsfunktionen Erfassung, Bearbeitung, Verwaltung und Speicherung von Dokumenten verstanden. [DMS08] S. 344.
Darunter fallen laut [DMS08] S. 3 folgende Punkte:

- Kennzeichnung und Beschreibung von Dokumenten (auch Metadaten des Dokuments genannt)
- Fortschreibung, Versionierung und Historienverwaltung von Dokumenten
- Ablage und Archivierung von Dokumenten
- Verteilung und Umlauf von Dokumenten
- Suche nach Dokumenten bzw. Dokumenteninhalten
- Schutz der Dokumente vor Verfälschung, Missbrauch und Vernichtung
- Langfristiger Zugriff auf die Dokumente und Lesbarkeit der Dokumente
- Lebenslauf und Vernichtung von Dokumenten
- Regelung von Verantwortlichkeiten für Inhalt und Verwaltung von Dokumenten

Der Begriff **„Dokumenten-Management-System“** muss auch in zwei verschiedene Sichtweisen differenziert werden:

„Bei **Dokumenten-Management-Systemen im engeren Sinne** geht es um die Logik der Verwaltung von Dokumenten, deren Status, Struktur, Lebenzyklus und Inhalt. Dokumente werden beschrieben, klassifiziert und in einer bestimmten logischen Struktur eingeordnet, damit sie einfach wieder gefunden werden können. Dokumente entstehen, werden verändert und (irgendwann) vernichtet.

Den **Dokumenten-Management-Systemen im weiteren Sinne** ordnet man auch noch weitere Funktionalitäten zu, wie z. B. Schrifterkennung, automatische Indizierung, [...], Publizierung. Hier lassen sich die Grenzen nicht mehr genau bestimmten!“

[DMS08] S. 5

Die Grundstruktur eines Dokumenten-Management-Systemes kann man dadurch grob in folgender Abbildung zusammenfassen:

Bild [DMS08] S. 38

Abbildung 2: Aufgabenbereiche eines Dokumenten-Management-Systems[2]

Dabei wird ein DMS-System in drei verschiedene Teilbereiche aufgegliedert:

4.2.1 Eingabe

Unabhängig des Ursprungs oder der Art des Dokumentes besitzt der Funktionsbereich Eingabe die Aufgabe diese Dokumente dem Dokumenten-Management-System zuzuführen. (Darunter fallen auch Post, Email, Andere anwendungen, fax usw. - Hier Hinweis auf WSDL)
Laut [DMS08] S. 40fff fallen in diesen Bereich zwei Funktionen:

[2]Quelle: [Barth08] S. 38

Dokumenteneingang Hier wird die Zuspielung der Dokumente in das DMS-System durch verschiedene Methoden behandelt / realisiert. Als mögliche Eingabe von Dokumenten kann sowohl das Einscannen von Textdokumente oder Bilder als auch der elektronische Eingang von Dokumenten durch E-Mail oder externen Anwendungen fungieren.
Auch hier gilt zu unterscheiden, dass durch den Einscannvorgang erstellte Dokumente als NCI-Dokument abgelegt werden und bereits digitalisierte Dokumente sich zur Umwandlung zu CI-Dokumenten anbieten. Sobald der Inhalt von eingescannten Dokumenten zur weiteren Verarbeitung ausgelesen bzw. ausgewertet werden soll, müssen die Dokumente in ein CI-Format transformatiert werden. Dies wird häufig durch eine OCR-Software realisiert, die beispielsweise das Bild eines eingescannten Briefes in (bearbeitbaren) Text umwandelt.
Bereits im CI-Format vorliegende Dokumente müssen nicht transformiert werden, jedoch werden die Dokumente oft in anderen Formaten zusätzlich abgespeichert. Ein Beispiel ist die Umwandlung eines Microsoft Word-Dokumentes in ein PDF-Dokument oder von unterschiedlichen Bildformaten in ein einheitliches Format.

Indizierung Bei der Indizierung werden Dokumente zur eindeutigen Identifikation mit Attributen versehen. Diese Attribute werden teilweise autonomisch durch das DMS-System anhand einer hochzählenden Identifikationsnummer oder manuell durch den Benutzer beim Einstellen des Dokumentes hinzugefügt. Solche Attribute werden auch als Metadaten des Dokumentes bezeichnet und meist als zusätzliche Suchkriterien angeboten.
Dabei werden in [DMS08] S. 44 zwei verschiedene Methoden zur automatischen Klassifizierung genannt. Beim wissenbasiertem Ansatz wird mittels umfangreichem Wissen über das Umfeld der Dokumente und dadurch abgeleitete Regeln dem System ermöglicht diese Dokumente automatisch einzuordnen und zu indizieren. Eine weitere Möglichkeit eröffnet sich durch das Verwenden von neuronalen Netzen. Hierbei wird durch die Vorarbeit eines

Menschen Beispiele geschaffen anhand welcher sich das System selbstständig (Auswahl)Kriterien erzeugt. Je mehr korrekte Beispiele vorgegeben werden, desto besser und zuverlässiger arbeitet die automatische Klassifizierung.

4.2.2 Verwaltung und Archivierung

Bei der **Verwaltung** werden die Probleme beim *Check-in* (Einspielen des Dokumentes), Bearbeitung und *Check-out* (Signalisierung der Weiterbearbeitung) behandelt, siehe auch Abbildung [DMS08] S. 38 ???. Wie auch bei einer Datenbank müssen Dokumente, die gerade bearbeitet werden, für andere Benutzer für Änderungen gesperrt werden, damit keine Inkonsistenzen auftretten können. Nach einer Bearbeitung und dem Check-in des abgeänderten Dokumentes muss die Versionsverwaltung des DMS-Systems beide Versionen beibehalten und (dabei) die ursprüngliche Version als veraltet und die neue Version als solche kennzeichnen. Zusätzlich muss die Wiederherstellung einer älteren Revision als aktuelles Dokument unterstütz werden.
Die **Archivierung** befasst sich mit der Sicherung und Wiederherstellung von Dokumenten und deren Metadaten. Im Zusammenhang mit DMS-Systemen springt man auch von einer revisionssicheren Archivierung. Dabei müssen laut [DMS08] S. 288 unter anderem bestimmte Punkte beachtet / eingehalten werden:

- Jedes Dokument muss unveränderbar archiviert werden.
- Es darf kein Dokument auf dem Weg ins Archiv oder im Archiv selbst verloren gehen.
- Kein Dokument darf während seiner vorgesehenen Lebenszeit zerstört werden können.
- Jedes Dokument muss in genau der gleichen Form, wie es erfasst wurde, wieder angezeigt und gedruckt werden können.

4.2.3 Ausgabe

Wie die Eingabe besteht die Ausgabe aus zwei Funktionen:

Recherche Die Recherche ist die Suche nach einem Dokument entweder durch eine strukturierte Suche anhand von zuvor eingetragenen Attributen (Author, Erstellungsdatum, Speichergröße usw.) oder durch eine Volltextsuche.
Die **strukturierte Suche** ist nur bei einer qualitativ hochwertigen Indizierung effizient, bietet dafür auch mit guter zeitlichen Performanz die besten / genauersten Ergebnisse, sofern die Indizierung entsprechend aufgebaut / eingehalten wurde.
Die **Volltextsuche** durchsucht „stupide“ den Inhalt der Dokumente nach den eingegebenen Suchbegriffen. Daher ist die Qualität der Suchergebnisse unabhängig von der Qualität der Indizierung. Jedoch können nur CI-Dokumente, deren Informationen auch durch den Computer auslesbar und interpretierbar sind, durchsucht werden. NCI-Dokumente wie Bilder oder Videos können durch die Volltextsuche nicht gefunden werden.

Reproduktion In diesem Teilbereich können die gespeicherten Dokumente wieder vom Benutzer abgerufen werden. Dies ist durch eine einfach Anzeige im Webbrowser, eine Weiterleitung per E-Mail oder eine Sendung als Druckauftrag möglich.

4.3 Content-Management-Systeme

Bei einem Content-Management-System (CMS) steht nicht mehr das eigentliche Dokument im Vordergrund, sondern vielmehr der enthaltene Informationsgehalt des Dokuments. Der Unterschied zwischen einem DMS und einem CMS besteht laut [DMS08] S. 114 im/in Folgenden/m:

> „Ein DMS hat als kleinestes Objekt der Betrachtung eines einzelnen Dokument. [...] Content-Management ist auf logische Infor-

> mationseinheiten ausgerichtet. Es ist z.B. das Ziel des Content-Managements, Inhalte, die auf mehrere Quellen verteilt sind, neue zusammenzustellen und daraus z.B. ein neues Dokument zu generieren."
>
> [DMS08] S. 114f

Die folgende Abbildung soll den (charakteristischen) Unterschied zwischen CMS-Systemen und DMS-Systemen verdeutlichen.

Bild [DMS08] S. 115

Wie zuvor beschrieben ist die Sichtweise eines DMS nur auf die einzelnen Dokumente beschränkt, während ein CMS einzelne Elemente / Informationen aus den Dokumenten extrahieren und ggf. zu einem neuen Dokument verschmelzen kann. Die Sichtweise des CMS wird durch das gestrichelte Polygon dargestellt, welches hier dokumentenübergreifend abgebildet ist.
Der (theoretische/beabsichtigte) Zweck, weshalb ein CMS-System eingesetzt wird, ist laut Oracle folgendermaßen definiert:

> „The key to a successful content management implementation is unlocking the value of content by making it as easy as possible for it to be consumed. This means that any piece of content must be available to any consumer, no matter what their method of access."
>
> [UCM07] S. 12

Ein CMS soll die Informationen jedes/jedwedem (Inhalts) extrahieren/aufnehmen und jedes Einzelteil / Element dieser Information den Benutzern zugänglich machen, unabhängig von der Art des Zugriffs. Dieses Konzept soll in Abbildung 3 verdeutlicht werden.

[3] Quelle: [UCM07] S. 12

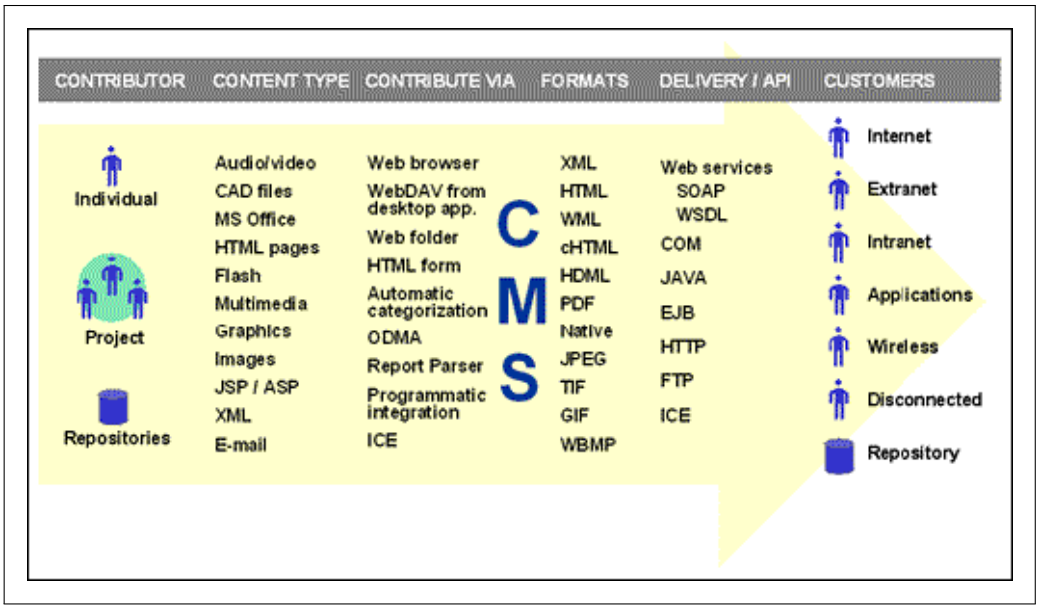


Abbildung 3: „any-to-any“ Content-Management Konzept[3]

Das CMS steht hier in der Mitte der Abbildung als Medium zwischen den verschiedenen Inhalten, eingestellt von den *Contributors* (links), und den Anwendern, die auf transformierte Versionen der Inhalte durch unterschiedliche Arten zugreifen (rechts).

4.4 [Enterprise-Content-Management-Systeme]

In diesem Zusammenhang / Kontext sei auch der Begriff Enterprise-Content-Management (ECM) genannt. Laut der „Association for Information and Image Management“ (AIIM[4]), welche sich mit umfasst dieser Begriff die Verwaltungfunktionen von Unternehmensinformationen in unterschiedlichen Dokumentformaten.[5] Diese Funktionen werden laut [DMS08] S. 116 durch verschiedene „Systeme wie Dokumenten-Management, Groupware, Workflow, Input- und Output-Management, (Web-)Content-management, Archivierung, Records-Management und andere“ bereitsgestellt.

[4]Die AIIM ist eine Gesellschaft von internationalen Herstellern und Anwendern von Informations- und Dokumenten-Mangement-Systemen
[5]Quelle: `http://www.aiim.org/What-is-ECM-Enterprise-Content-Management.aspx`

4.5 Universal-Content-Management-Systeme

Im Gegensatz zu anderen CMS-Systemen, wie Typo3 oder Joomla, bezeichnet Oracle seine Softwarelösung in diesem Bereich als Universal-Content-Management (UCM). Jedoch unterscheidet es sich in der technisches Umsetzung nicht von anderen CMS-Systemen. Es wir vermutet, dass sich Oracle durch diese Bezeichnung von den anderen CMS-Systemem abheben / absetzten wollte, also nur aufgrund von Marketing-Vorteilen ihr Produkt so nannte.

4.6 Service-Orientierte Architektur

Eine eindeutige und einheitliche Definition einer Service-Orientierter Architektur (SOA) existiert nicht. Einen Versuch einer Definition wird in [SOA07] beschrieben:

> „[...] a service oriented architecture is an architecture for building business applications as a set of loosely coupled black-box components orchestrated to deliver a well-defined level of service by linking together business processes.“
>
> [SOA07] S. 27

SOA ist ein Ansatz im Bereich der Informationstechnik um Anwendungen oder einzelne Dienste aus verschiedenen Geschäftsprozessen zu bilden. Melzer bietet eine ausführlichere Definition:

> „Unter einer SOA versteht man eine Systemarchitektur, die vielfältige, verschiedene und eventuell inkompatible Methoden oder Applikationen als wiederverwendbare und offen zugreifbare Dienste repräsentiert und dadruch eine plattform- und sprachenunabhängige Nutzung und Wiederverwendung ermöglicht. “
>
> [Melzer08] S. 13

Zur Verdeutlichung einer SOA kann ein beispielhafter und vereinfachter Aufbau eines Online-Shops verwendet werden.

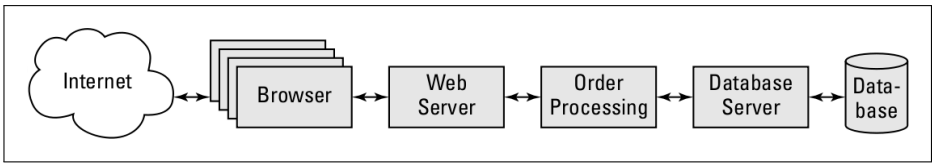


Abbildung 4: Simple Software Architektur eines Webshops[6]

Durch den gewöhnlichen Browser können Benutzer auf die Webseite des Webservers zugreifen um dort auf die eigentliche Anwendung des Webshops *Order Processing* zuzugreifen. Dabei werden durch einen Datenbankserver die Informationen in einer Datenbank gespeichert oder von dort der Webshop-Anwendung zugänglich gemacht. Welche Funktion die Anwendung *Order Processing* ausführt hängt von den Aufforderungen des Benutzers durch den Browser ab.
Dieser Struktur wird nun ein Service-Orientierte Komponente *Credit Checking* hinzugefügt, siehe Abbildung 5.

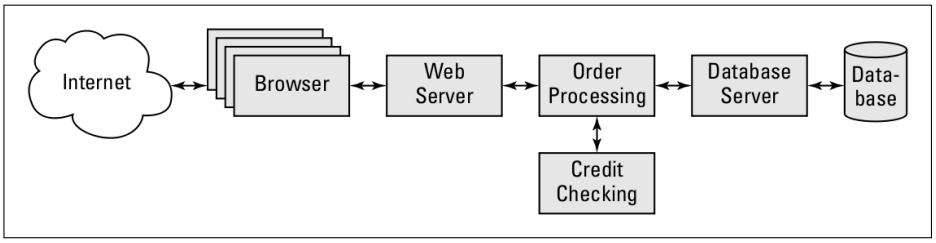


Abbildung 5: Erweiterte Software Architektur mit Service-Orientierte Komponente[8]

Dabei hat die eigentliche Anwendung des Webshops keine Kenntniss wie die Komponente *Credit Checking* intern abläuft, sondern übergibt nur die essentiellen Informationen, in diesem Fall die Kreditkartendaten, an die Komponente. Es ist unrelevant, ob diese Komponenten eine externe Datenbank oder Webseite nach der Kreditwürdigkeit des Benutzers befragen, solange die Komponente auswertbare Informationen (zahlungsfähig ja/nein) an die

[6]Quelle: [SOA07] S. 18
[8]Quelle: [SOA07] S. 20

Webshop Anwendung liefert. Für die Anwendung *Order Processing* ist die Komponente *Credit Checking* eine sogennante **black box**.
Die komplexen Berechnungen und Algorithmen zur Bestimmung der Kreditwürdigkeit des Benutzers werden komplett verdeckt, so dass nur die Kreditkarteninformationen der Komponente zu übergeben sind.
Die Komponente *Credit Checking* steht der Webshop Anwendung als **abstrahierter Dienst bzw. Service** zur Verfügung.

4.7 Web-Services-Architektur

Wie bei dem Begriff SOA gibt es für Webs Services keine allgemein gültige Definition, jedoch überlappen sich Definitionsvorschläge in verschiedenen Gesichtspunkten. Laut Melzer ([Melzer08] S. 55) bietet das World Wide Web Consortium (W3C) den konkretesten Ansatz einer passenden Definition.

> „A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.“
>
> [W3WS04] S. 7

Ein Web Service ist so aufgebaut, dass ein Zusammenspiel zwischen Rechner über ein Netzwerk möglich ist. Dabei ist Schnittstelle des Web Services in einem maschinell interpretierbaren Format gehalten, so dass andere Systeme auf diese Schnittstelle zugreifen können. Dieser Zugriff findet durch das Simple Object Access Protocol (SOAP) statt, welches üblicherweise über das Hypertext Transfer Protocol (HTTP) versendet wird. Die SOAP-Nachrichten sind nach dem XML-Schema zusammen mit anderen Web-Standards aufgebaut.

Daraus leitet Melzer folgende Spezifikationen für eine Web-Services-Architektur ab:

SOAP beschreibt das XML-basierte Nachrichtenformat der Kommunikation und dessen Einbettung in ein Transportprotokoll.

WSDL ist eine - ebenfalls XML-basierte - Bschreibungssprache, um Web Services (Dienste) zu beschreiben.

UDDI beschreibt einen Verzeichnisdienst für Web Services. UDDI (Universal Description, Discovery and Integration protocol) spezifiziert eine standardisierte Verzeichnisstruktur für die Verwaltung von Web-Services-Metadaten. Zu den Metadaten zählen allgemeine Anfornderungen, Web-Services-Eigenschaften oder die benötigten Informationen zum Auffinden von Web Services.

[Melzer08] S. 55

Dabei erwähnt Metzer ([Melzer08] S. 56), dass ein Verzeichnisdienst keine Notwendigkeit für die Verwendung eines Web Services ist, „sondern vielmehr die Infrastruktur zum Auffinden von geeigneten Web Services beschreibt.“ Der Ablauf der Benutzung eines Web Services soll durch folgende Abbildung verdeutlicht werden:

Quelle: [Melzer08] S. 56

Abbildung 6: Ablauf einer Web Service Benutzung[9]

1. Der Anbieter des Web Services muss seinen Dienst durch eine WSDL-Datei in Form einer XML-Datei dem Diensverzeichnis bekannt geben.

2. Erst dann können mögliche Nutzer dieses Dienstes den Web Service im UDDI-basiertem Diensverzeichnis finden. Die Suchanfrage findet über eine SOAP-Schnittstelle statt.

[9]Quelle: [Melzer08] S. 56

3. Ein Verweis auf den Dienst in Form einer WSDL-Datei wird an den Dienstbenutzer als Antwort der Suchanfrage gesendet.

4. Durch diesen Verweis erfährt der Benutzer die Adresse des Dienstanbieters und kann die Beschreibung des Web Services abfragen.

5. Nach Erhalt dieser Beschreibung kann der eigentliche Webdienst mittels SOAP verwendet werden.

5 Nagios

5.1 Allgemein

Nagios dient zum Überwachen von Hosts und deren Services in komplexen Infrastrukturen(Host und Services erklären?) und wurde von dem Amerikaner Ethan Galstad seit 1999[10] - damals unter der Vorgängerversion NetSaint - entwickelt und bis heute gepflegt. Galstad gründete aufgrund der vielfältigen(ansturmmäßig) und positiven Resonanz am 9. November 2007 die „Nagios Enterprises LLC“, welche Nagios als kommerzielle Dienstleistung anbietet. Die Software selbst blieb weiterhin unter der freien Lizenz „GNU General Public License version 2“[11] verfügbar. Diese erlaubt Einblick in den Programmcode und das Modifizieren der Anwendung nach eigenen Vorstellungen.

Nagios erfreut sich hoher Beliebtheit aufgrund der (bereits vorhandenen [macht kein sinn hohe beliebtheit aufgrund der großen community?]) großen Community, die Tipps, Ratschläge und auch eigene Nagios-Plugins kostenlos anbietet. Außerdem können selbst mit geringen Programmierkenntnissen zusätzliche Skripte zur Überwachung geschrieben werden, wenn ein spezieller Anwendungsfall dies erfordert. Warum wird Nagios engesetzt und nicht was anders -> andere kandidaten finden openview, big brother? -> das buch vom jäger verwenden!

OpenSource halt, recht einfach plugins programmierbar (auf plugin kapitel verweisen)

[10] Quelle: `http://www.netsaint.org/changelog.php`
[11] Quelle: `http://www.gnu.org/licenses/old-licenses/gpl-2.0.txt`

5.2 Aufbau / Architektur

Barth schreibt über Nagios:

> „Die große Stärke von Nagios - auch im Vergleich zu anderen Netzwerküberwachungstools - liegt in seinem modularen Aufbau: Der Nagios-Kern enthält keinen einzigen Test, stattdessen verwendet er für Service- und Host-Checks externe Programme, die als *Plugins* bezeichnet werden.“
>
> [Barth08] S. 25

Dieser „Kern“ beinhaltet das komplette Benachrichtigungssystem mit Kontaktadressen und Benachrichtigungsvorgaben (Zeit, Art, zusätzliche Kriterien), die Hosts- und Servicedefinitionen inklusive deren Gruppierungen und schließlich das Webinterface. Die eigentlichen Checks in Form der selbständigen Plugins sind abgekapselt von diesem Kern. Siehe hierfür auch Abbildung 7.

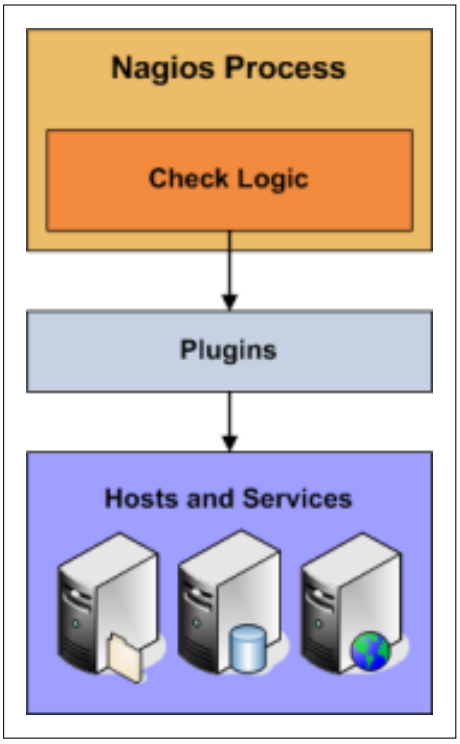


Abbildung 7: Aktive Checks durch Nagios[12]

Damit Nagios die gewünschten Server überwachen kann, müssen sie der Anwendung zuerst bekannt gemacht werden. Dies wird über das Anlegen einer Konfigurationsdatei mit einem Host-Objekt erreicht. Dabei richtet sich die Definition des Host-Objektes nach dem Schema, welches für alle Objektedefinitionen (Services, Kontakt, Gruppen, Kommandos etc.) gilt:

```
1 define object-type {
2         parameter value
3         parameter value
4         ...
5 }
```

Listing 1: Nagiosschema für Objektdefinitionen

Eine gültige Host-Definition muss mindestens folgende Elemente besitzten:

```
1  define host{
2          host_name                 example.kit.edu #Referenzname des Servers
3          alias                     Oracle UCM Server #Weitere Bezeichnung
4          address                   example.kit.edu #FQDN des Rechners
5          max_check_attempts        4 #Anzahl der Checks zum Wechsel von Hard-
               zu Soft-State
6          check_period              24x7 #Zeitraum der aktiven Checks
7          contact_groups            UCM-admins #Zu alarmierende Benutzergruppe
8          notification_interval     120 #Minuten bis Alarmierung wiederholt wird
9          notification_period       24x7 #Zeitraum der Benachrichtigungen
10 }
```

Listing 2: Definition eines Hostobjektes

Gewöhnlich / In der Praxis werden öfters verwendete Attribute wie die Kontaktgruppe oder der Zeitraum für die aktiven Checks durch Verwendung eines übergeordneten Host-Objektes nach unten vererbt. Dadurch müssen nurnoch die spezifischen Inforamtionen und der Name des übergeordneten Host-Objektes eingetragen werden.

```
1 define host {
2         use               windows-server #Oberklasse dieses Host-Objektes
3         host_name         example.kit.edu
4         alias             Oracle UCM Server
5         address           example.kit.edu
6 }
```

Listing 3: Verkürzte Definition eines Hostobjektes

Mit dieser Hostdefinition wird der Rechner im Webinterface von Nagios bereits angezeigt:

Jedoch wird nur die Erreichbarkeit über das Netzwerk mit einem Ping überwacht. Um andere Dienste zu überwachen müssen die gewünschten Plugins

[12] Quelle: `http://www.nagioswiki.org/w/images/8/81/Plugins.png`

Host	Service	Status	Last Check	Duration	Attempt	Status Information
example.kit.edu	PING	OK	2009-07-23 13:47:24	0d 0h 3m 15s	1/4	PING OK - Packet loss = 0%, RTA = 26.66 ms

Abbildung 8: Anzeige des Servers im Webinterface von Nagios

explizit aus dem Nagios Repertoire dem zu überwachendem Computer mit einem ähnlichen Schema zugeteilt werden.

Eine beispielhafte Servicedefinition für die Überwachung des Webservers auf dem Host *example.kit.edu* wird in Codelisting 5 gezeigt.

```
1 define service{
2         use                     generic-service #Oberklasse dieses Service-
              Objektes
3         host_name               example.kit.edu
4         service_description     HTTP Server #Bezeichnung des Checks
5         check_command           check_http #Angabe des NagiosPlugins (hier
              ohne Parameter)
6         }
```

Listing 4: Verkürzte Definition eines Hostobjektes

Die Plugins werden durch die Servicedefinitionen mit den jeweiligen Hosts verbunden und durch das Attribut *check_command* mit ggf. veränderten Parametern durch Nagios aufgerufen.

Nagios wird in einem festlegbarem / veränderbarem Zeitintervall alle vom Benutzer definierten Host- und Servicechecks überprüfen und die Ergebnisse der entsprechenden Plugins verarbeiten / auswerten.

Weiterhin beschreibt Barth die Plugins folgendermaßen:

> „Jedes Plugin, das bei Host- und Service-Checks zum Einsatz kommt, ist ein eigenes, selbständiges Programm, das sich auch unabhängig von Nagios benutzen lässt."
>
> [Barth08] S. 105

Daher lassen sich die Parameter eines Plugins folgendermaßen überprüfen:

```
paul@iwrpaul:/usr/lib/nagios/plugins$ ./check_swap -w 20 -c 10
SWAP OK - 96% free (1826 MB out of 1906 MB) |swap=1826MB;0;0;0;1906
```

Abbildung 9: Beispielhafte manuelle Ausführung eines Servicechecks

Die Ausgabe des Plugins gibt den Zustand des Services an; in diesem Fall wird kein Schwellwert überschritten, daher die Meldung „`SWAP OK`“. Dieses Plugin liefert noch zusätzliche Performance-Informationen, die mit externen Programmen ausgewertet, gespeichert und visualisiert werden können. Standardmäßig werden die Performancedaten von der normalen Ausgabe mit einem „|“ getrennt. Jedoch können auch Werte aus der normalen Textausgabe verwendet werden, so dass in diesem Beispiel keine Berechnung des Prozentsatzes notwendig wäre.

Um diesen Service mit den angegebenen Schwellwerten von Nagios überwachenzulassen, muss folgende Servicedefinition in die Konfigurationsdatei eingetragen werden:

```
1  # Define a service to check the swap disk space
2  # on the local machine.  Warning if =< 20% free,
3  # critical if =< 10% free space on swap partition.
4
5  define service{
6   use          generic-service
7   host_name    example.kit.edu
8   service_description  Swap Disk Space
9   check_command    check_swap!-w 20% -c 10%       # Angabe des zu verwendenden
        Plugins mit WARNING (respektiv) CRITICAL Schwellwertparameter
10 }
```

Listing 5: Beispielhafte (Definition) eines Servicechecks

Dabei wird in vier verschiedene Rückgabewerte / Antworten der Plugins unterschieden:

Status	**Bezeichnung**	**Beschreibung**
0	OK	Alles in Ordnung
1	WARNING	Die Warnschwelle wurde überschritten, die kritische Schwelle ist aber noch nicht erreicht.
2	CRITICAL	Entweder wurde die kritische Schwelle überschritten oder das Plugin hat den Test nach einem Timeout abgebrochen.
3	UNKNOWN	Innerhalb des Plugins trat ein Fehler auf (zum Beispiel weil falsche Parameter verwendet wurden)

Tabelle 1: Rückgabewerte für Nagios Plugins[13]

Anhand dieser Werte wertet Nagios gezielt den Status des jeweiligen Objektes (Host oder Service) aus. Weiterhin gibt es weiche (Soft States) und harte Zustände (Hard States):

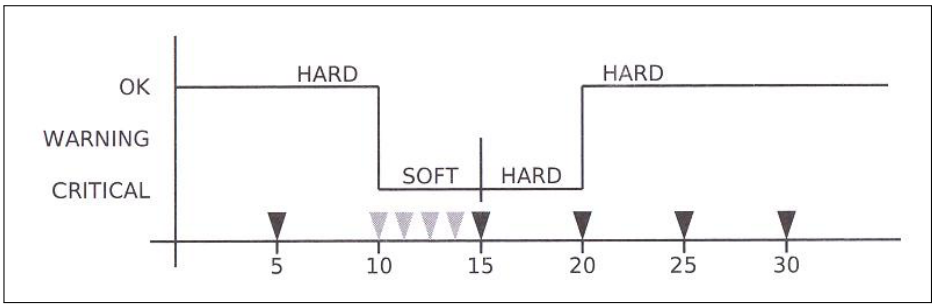


Abbildung 10: Beispiel für den zeitlichen Verlauf durch vers. Zustände[14]

Ausgehend von einem OK Zustand wird in diesem Beispiel jede fünf Minuten periodisch überprüft, ob sich der Status des überwachten Objektes verändert hat. Nach zehn Minuten wird ein Umschwenken / Änderung des Zustandes durch das jeweilige Plugin gemeldet. Hier im Beispiel wechselt der Zustand nach CRITICAL, zunächst allerdings als Soft State Daher wird durch Nagios noch keine Benachrichtigung versendet, da es sich um eine Falschmeldung, auch False Positive genannt, handeln kann. Aufgrund einer kurzweiligen / kurzfristigen (besseres Wort? peak mäßig) hohen Auslastung des Netzwerkes oder um ein kurzzeitiges Problem, welches sich von alleine wieder normalisiert. (Bspw. Prozessorauslastung)
Um einen False Positive auszuschliessen, wird der im Soft State befindliche Service bzw. Host mit einer höheren Frequenz überprüft. Sollten diese Überprüfungen den vorherigen Zustand bestätigen, verfestigt sich der aktuelle Zustand, man spricht nun von einem Hard State / wechselt der Zustand in den Hard State. Erst in diesem Moment werden die entsprechenden Kontaktpersonen über den in diesem Beispiel kritischen Zustand benachrichtigt. Sollte sich der Zustand wieder in den Normalzustand begeben und dieser Zustandsübergang wird von dem (von Nagios ausgeführten) Plugin festgestellt, wird dies an den Nagios Server gemeldet.

[13]Quelle: [Barth08] S. 105f
[14]Quelle: [Barth08] S. 95

Ein Übergang zu dem OK Status wird sofort als Hard State festgelegt / festgehalten / festgesetzt / und führt dadurch zur sofortigen Benachrichtigung durch Nagios.
Diese Benachrichtigung

- Betroffene OSI Schichten auflisten und erklären
- Wie werden die Info von Nagios gesammelt und wie gespeichert -> FlapDetection
- FLapping [15]
- Benachrichtigung durch email oder sms sogar per Telefon geht usw.
- Hierarchie `http://nagios.sourceforge.net/docs/3_0/networkreachability.html`

5.3 Überprüfungsmethoden

Dienste, die im Netzwerk zur Verfügung stehen (Netzwerkdienste), wie ein Web- oder FTP-Server , lassen sich einfach / simpel direkt über das Netz auf ihren Zustand (hin) überprüfen / testen. Hierfür muss dem entsprechende Plugin lediglich die Netzwerkadresse mitgeteilt werden, siehe Abbildung 11 als beispielhafte Überprüfung eines Webservers.

```
liwrpaul:/usr/lib/nagios/plugins$ ./check_http -H scc-bm-01.scc.kit.edu
OK HTTP/1.1 200 OK - 5194 bytes in 0.008 seconds |time=0.008195s;;;0.000000 size=5194B;;;0
```

Abbildung 11: Beispielhafte manuelle Ausführung eines netzwerkbasierenden Servicechecks / HTTP Server Check

(Bitte beachten, dass das Plugin immernoch auf dem Nagios Server ausgeführt wird / sich immernoch auf dem Nagios Server befindet)
Dienste, die sich nicht standardmäßig / ohne weiteres / ohne weitere Anpassung(en) über das Netzwerk überprüfen lassen, wie die Kapazität einer

[15] `http://nagios.sourceforge.net/docs/2_0/flapping.html`

Festplatte auf einem entfernten Server(, das (Laufen) eines Prozesses) oder die Durchsuchung einer Logdatei nach bestimmten (Stop)wörtern.
Nagios bietet verschiedene Möglichkeiten an solche Dienste zu überprüfen:

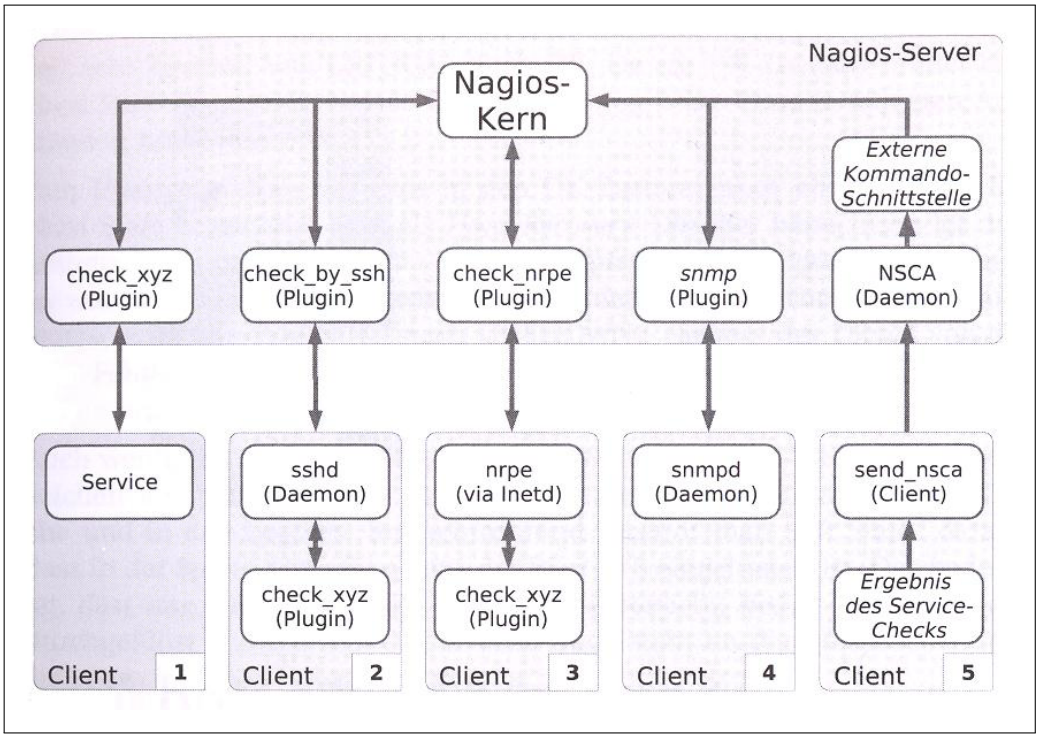


Abbildung 12: Verschiedene Überwachungsmöglichkeiten von Nagios[16]

Man unterscheidet generell zwischen aktiven und passiven Checks.

5.3.1 Aktive Checks

asdaasdadasdasdads asdasd

asdasdas

5.3.2 Passive Checks

asdasdasdasdas

Methode 1 - Netzwerkdienste Der zuvor, in Abbildung 11, gezeigte Test eines netzwerkbasierenden Dienstes wird im obigen Bild mit dem Client-Rechner (mit der Nummer) 1 abgebildet. Dies ist die einfachste Überwa-

[16]Quelle: [Barth08] S. 98
[16]Quelle: `http://nagios.sourceforge.net/docs/3_0/images/activechecks.png`

chungsmethode, da keine zusätzlichen Programme oder aufwändige Konfiguration benötigt wird. Vorteilhaft ist auch, dass der Dienst über das Netzwerk getestet wird, so wie der Benutzer auch auf den Dienst zugreift. Damit können auch gleichzeitig andere Knotenpunkte wie Switches überwacht werden.

Methode 2 - SSH Falls es sich beim Client um ein Unixderivat handelt, ist der entfernte Zugriff auf diesen Client per SSH[17]-Dienst möglich. Dazu muss auf dem Client ein SSH-Benutzerkonto angelegt sein, mit dem sich Nagios anmelden kann und die öffentlichen Schlüssel (zwischen Nagios Server und Client) ausgetauscht werden, damit keine passwortabhängige Benutzerauthentifizierung (Eingabe von PW) notwendig ist. Danach können lokale Ressourcen, wie Festplattenkapazität oder Logdateien mit dem entsprechenden Plugin direkt auf dem entfernten Rechner überwacht werden. Damit der Client diese Plugins verwenden kann, müssen sich die gewünschten Plugins (auch) auf dem Client (lokal) befinden. Eine beispielhafte Verwendung mit dem dafür gedachten Nagios Plugin „`check_by_ssh`“ (von dieser Überwachungsmethode) wird in Abbildung 13 gezeigt.

```
paul@iwrpaul:/usr/lib/nagios/plugins$ ./check_by_ssh -H ppt.ka.fzk.de -C "/bin/check_swap -w 20 -c 10"
SWAP OK - 100% free (384 MB out of 384 MB) |swap=384MB;0;0;0;384
```

Abbildung 13: Beispielhafte manuelle Ausführung eines Servicechecks über SSH

(Hier beachten, dass kein Passwort abgefragt wird, daher zuvor Schlüsselaustauschen)

Methode 3 - NRPE Eine alternative Möglichkeit solche Dienste auf entfernten Rechnern zu überwachen, ist durch den sogenannten Nagios Remote Plugin Executor (NRPE). Hier muss auf dem Client ein „Agent“ installiert werden, welcher einen Port öffnet mit dem der Agent mit dem Nagios Server kommuniziert.

[17] Durch eine Secure Shell (SSH) kann man sich eine verschlüsselte Netzwerkverbindung zum entfernten Rechner aufbauen.

[18] Quelle: `http://www.nagios.org/images/addons/nrpe/nrpe.png`

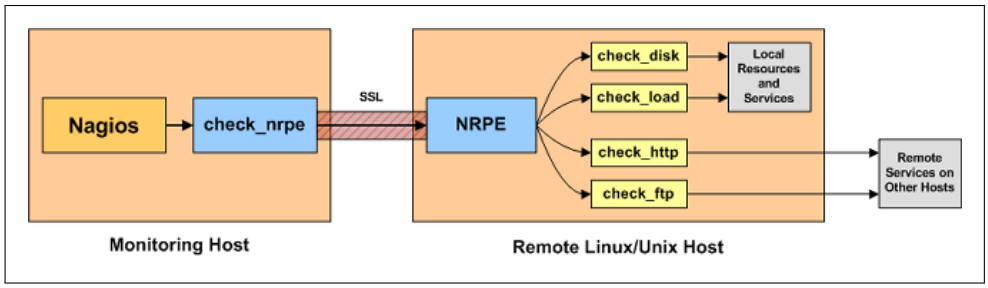


Abbildung 14: Aktive Checks mit NRPE[18]

Der Nagios Server kann dann Anforderungen über das Nagios-Plugin „`check_nrpe`“ an den Client verschicken. Ein Aufruf dieses Plugins ist dem des „`check_by_ssh`“ Plugins, siehe dazu Abbildung 13, sehr ähnlich.
Der Nachteil dieser Variante ist ein zusätzlich geöffneter Port und der höhere / erhöhte Aufwand beim Installieren des Agenten im Gegensatz zum (vermutlich /meistens) bereits laufendem SSH-Dienst. Zusätzlich gibt es nur die Möglichkeit die Anfragen auf diesem Port auf bestimmte IPs zu beschränken, jedoch nicht den Zugriff durch ein Passwort zu sichern. Dafür beschränkt sich der NRPE (lediglich) auf die auf dem entfernten Client liegenden Nagios Plugins und kann nicht System- bzw. Benutzerkommandos aufrufen, wie bspw. das „`rm`“ Kommando zum Löschen von Dateien, welche durch den Einsatz von „`check_by_ssh`“ standardmäßig möglich wären. Sicherheitstechnisch gesehen ist die SSH-Variante kritischer, da es einem Angreifer ermöglicht auf diese System- bzw. Benutzerkommandos zuzugreifen, wenn er die Kontrolle über den Nagios Server erlangt. Beide Verfahren unterstützen die Verschlüsselung der Datenübertragung zwischen Nagios-Server und Client, so dass keine Informationen im Klartext übertragen werden.

Methode 4 - SNMP Diese Variante wird nur verkürzt behandelt, da sich diese Arbeit hauptsächlich mit der Überwachung von Servern beschäftigt und nicht von Netzwerkkomponenten wie Switches oder Router, die nur durch das Simple Network Management Protocol (SNMP) überwacht werden können, wenn mehr Informationen als eine schlichte Erreichbarkeit überprüft / gesammelt werden soll.

Barth schreibt über diese Variante / Überwachungsmethode:

> „Mit dem Simple Network Management Protocol SNMP lassen sich ebenfalls lokale Ressourcen übers Netz abfragen [...]. Ist auf dem Zielhost ein SNMP-Daemon installiert [...] kann Nagios ihn nutzen, um lokale Ressourcen wie Prozesse, Festplatten oder Interface-Auslastung abzufragen."
>
> [Barth08] S. 101

Durch SNMP kann auf die strukturierte Datenhaltung der MIB[19] in den entfernten Netzwerkknoten zugegriffen werden. Die MIB-Struktur ist folgendermaßen aufgebaut:

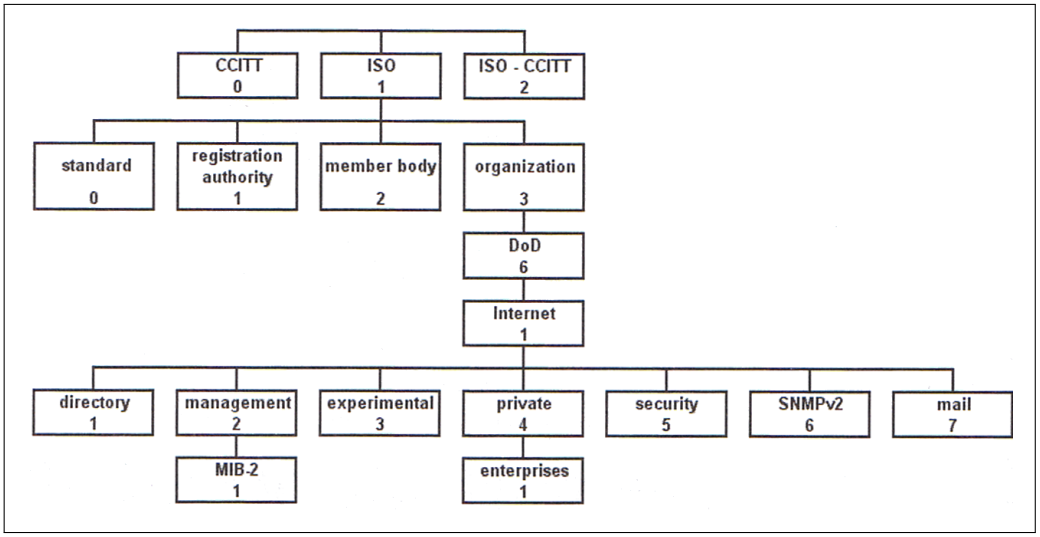


Abbildung 15: Struktur der Management Information Base[20]

Anhand dieser Anordnung können die SNMP-Plugins von Nagios den gewünschten Wert über das Netzwerk abfragen. Bei einem Switch werden die auslesbaren Informationen vom Hersteller bestimmt. Wenn auf einem Rechner eigene Ergebnisse in der MIB abgespeichert werden sollen, muss dies

[19] Die Management Information Base (MIB) dient als SNMP-Informationstruktur und besteht aus einem hierarchischen, aus Zahlen aufgebauten Namensraum. Ähnliche Struktur wie andere hierarchische Verzeichnisdiensten wie DNS oder LDAP. Quelle: [Barth08] S.233

[20] Quelle: [Munin08] S. 156

durch einen SNMP-Daemon eingetragen werden. Dessen Konfiguration ist im Vergleich zu den anderen Überwachungsmethoden deutlich komplexer.
Es gibt zwei verschiendene Möglichkeiten Dienste mit SNMP zu überwachen. Der Server frägt aktiv den Inhalt der entsprechenden MIB Einträgen periodisch ab oder der Client sendet asychron seine Statusmeldungen an den Nagios-Server. Beim letzteren spricht man auch von so genannten SNMP-Traps.

- Warum wird SNMP nicht verwendet?
- klartextübertragung bis SNMP 2c
- Schreibrechte können Schaden anrichten
- Brute Force attacken ausgesetzt
- Beschränkte Ausgabemögliochkeit / maximale Datengrösse der Ausgabe -> Logüberwachung nur mit Aufwand möglich autarkes Programm von Nöten, dann muss selbst das Ergebniss in die MIB geschrieben werden, damit Nagios darauf Zugriff erlangt -> zu aufwändig im Vergleich mit Agenten

Methode 5 - NSCA Diese Methode verwendet passive Checks. Bei passiven Tests führt der zu überwachende Computer das statuserzeugende Plugin selbst aus und sendet es über ein weiteres Plugin zum Nagios-Server. Hierfür muss das Testprogramm bzw. Script und das entsprechende Plugin „`send_ncsa`“, welches zum Versenden der Informationen zuständig ist, auf dem Host vorhanden sein. Auf der anderen Seite muss der „`NSCA`“ (Nagios Service Check Acceptor) auf dem Nagios-Server als Daemon gestartet sein, damit die übermittelten Ergebnisse von Nagios entgegengenommen werden.
Folgende Abbildung soll das Prinzip der passiven Checks verdeutlichen:
Das Testprogramm *Remote Application* wird selbständig vom zu überwachenden Rechner *Remote Host* aufgerufen und übermittelt durch das „`send_ncsa`“

[21] Quelle: `http://www.nagios.org/images/addons/nsca/nsca.png`

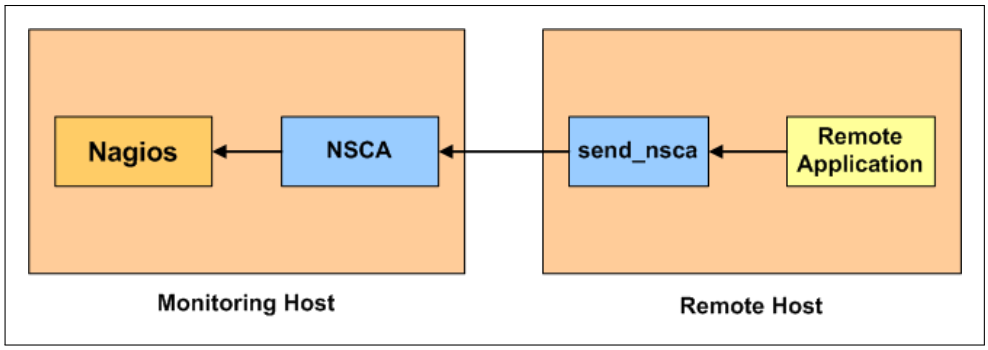


Abbildung 16: Passive Checks mit NSCA[21]

Plugin die Ergebnisse über das Netzwerk an den Nagios-Server *Monitoring Host*. Da auf diesem der NSCA als Daemon läuft können die Ergebnisse an die Nagios-Anwendung zur Auswertung weitergegeben werden.

- Kurz agenten, zeigen auf f. Kapitel -> SNMP erklären (MIB, OID) Sicherheitsrisiko

5.4 Überwachungslogik (mit Alarmierung/Benachrichtigung)

TODO: Distributed Monitoring bezug auf allg überwachungssysteme

5.5 Plugins

Gedacht für Linux umgebung

Verschiedene Möglichkeiten Checks zu realisieren unter Unix Systemen:

Leicht programmierbar -> perl Extra Plugins für Windows

5.6 (Windows) Agenten oder allgemein Einholen von Infos

Warum nicht einfach alles über SNMP? -> ODI muss man erst beantragen, hoher Aufwand und dann doch nicht so universell/alles abdeckend wie aktive checks, man kann keine logfiles durchuchen -> könnte es aber als standalone prog auf dem client laufen lassen und dieser sendet dann passive checks

[22]Quelle: `http://www.kilala.nl/Sysadmin/index.php?id=708`

	SSH	NRPE	SNMP	SNMP traps	NSCA
Connection initiation	Srv -> Clnt	Srv -> Clnt	Srv -> Clnt	Clnt -> Srv	Clnt -> Srv
Security	Encryption TCP wrappers Key pairs	Encryption Access List TCP wrappers	Access List (v2) Password (v3)	Access List (v2) Password (v3) TCP wrappers	Encryption Access List TCP wrappers
Configuration	On server	On client	On client	On client and On server	On client
Difficulty	Easy	Moderate	Hard	Hard	Moderate

Abbildung 17: Übersicht der verschiedenen Unix Agenten[22]

Sagen das auf alten NSClient verzichtet wird und OpMon Agent nicht behandelt

1. Bilder ausm Nagios Buch Seite 472ff!
2. NSClient++
3. NC_Net
4. NRPE_NT

Zusammenfassung?
Welche wird jetzt eingesetzt und warum?
Erwähne sichheitsstechnisch Parameter erlauben oder nicht erlauben
Dabei sagen, dass wenn nicht erlaubt sind keine zentrale Konfiguration der Checks auf dem Nagios server möglich ist -> abwägen

5.7 Visualisierung der eingesammelten Daten

	NSClient	NRPEnt	NSClient++	SNMP	SNMP traps	NC_net **
Connection initiation	Srv -> Clnt	Srv -> Clnt	Srv -> Clnt	Srv -> Clnt	Clnt -> Srv	Clnt -> Srv Srv -> Clnt
Security	Password	Password Encryption	Password Encryption * ACL	Access List Password	Access List Password	Encryption ACL
Configuration	On client	On client	On client	On client	On client and On server	On client
Difficulty	Moderate	Moderate	Moderate	Hard	Hard	Moderate
Resource usage ***	unknown	unknown	9MB RAM	unknown	unknown	30MB RAM

Abbildung 18: Übersicht der verschiedenen Windows Agenten
[23]

6 Oracle UCM

6.1 Allgemein

Oracle Universal Content Management basiert auf der Software Stellent von der gleichnamigen Firma Stellent, welche im November 2006[24] von Oracle gekauft / erworben wurde.

Quelle: [Huff06] S. 17

Abbildung 19: Beispielhafter Einsatz eines Content Servers[25]

Sessionanzahl

6.2 Aufbau / Architektur

Die Architektur des Oracle UCM Systems gliedert sich in separate Komponenten auf wie in Abbildung 20 gezeigt.

[24]Quelle: [OraPress]
[25]Quelle: [Huff06] S. 17
[26]Quelle: `http://www.club-oracle.com/forums/oracle-universal-content-management-ucm-aka-stellent`

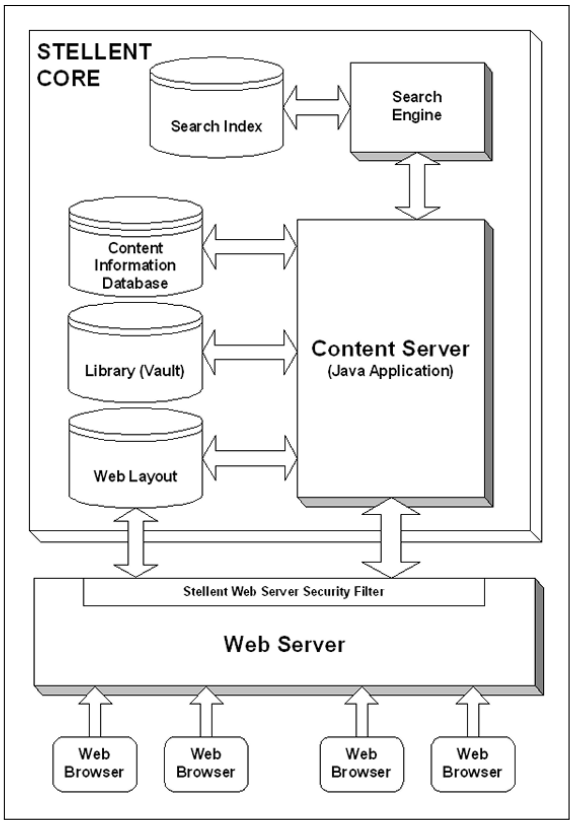


Abbildung 20: Oracle UCM Architektur[26]

6.2.1 Content Server

Der Content Server ist das Herzstück der Oracle UCM Anwendung. Er dient als Grundgerüst (Framework) für darüberliegende Funktionen, da er für die Ablage der Dokumente sowie der Verwaltung der drei Teilbereiche, siehe Abbildung 2, verantwortlich ist.

Dieses Framework ist als Service-Oriented Architecture (SOA) aufgebaut. Im Kontext des Content Servers wird als Service ein diskreter Aufruf einer Funktion verstanden. Dabei kann diese Funktion das Hinzufügen, die Bearbeitung, die Konvertierung oder das Herunterladen eines Dokumentes bedeuten. Diese Services und ihre einzelnen Funktionen werden durch das SOA-Framework verdeckt und stehen durch höhere abstrahierte Dienste zur Verfügung.

6.2.2 Vault und Web Layout

6.2.3 Inbound Refinery

6.2.4 Search Engine

6.2.5 Webserver

Für was wird sie im FZK benutzt, auch Windows nennen

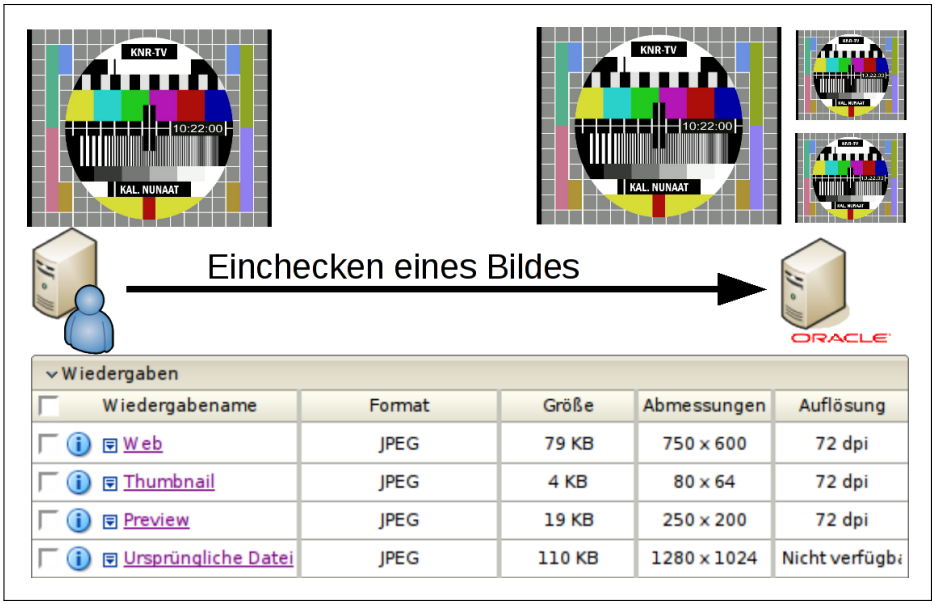


Abbildung 21: Bilddatenbank als Anwendung

7 Überwachungselemente

Die Überwachung einer Dienstes über ein Netzwerk verteilt sich auf verschiedenen Ebenen mit unterschiedlichen Gewichtungen. Zum Beispiel stellt das simple Senden eines Pings an den entsprechenden Server die niedrigste und primitivste Stufe dar, da hier lediglich die Netzwerkschnittstelle des Servers auf ihre Funktionalität und dabei der Status der Netzwerkstrecke getestet wird.(Rechner an und Netzwerk ok) Ob die Anwendung überhaupt auf dem Server läuft und wenn, in welchem Zustand sie sich befindet (betriebsfähig, reagiert nicht mehr usw.), muss auf eine andere Weise herausgefunden werden.

Dabei lässt sich aus den verschiedenen Überwachungselemente folgende drei Kategorien bilden/ableiten:

7.1 Statusabfragen

Diese Kategorie besteht aus einfacheren Überprüfungen, die jeweils den Status des Überwachungselementes überwachen. Dabei können weitere Untergruppen gebildet werden.
(Irgendwie Hierarchie verdeutlichen: zuerst ping als Grundlage, dann prozesse und services dann funktionschecks, bloss wie? Visio Bild?)

System

- **Ping** Überprüft, ob der Rechner vom Nagios-Server über das Netzwerk erreichbar ist.
- **Prozessorauslastung** Überwacht die Auslastung des Prozessors und schlägt bei ungewöhnlich hohen Werten Alarm.
- **Festplattenspeicherausnutzung** Überwacht die Speicherplatzauslastungen der verschiedenen Festplattenpartitionen, damit immer genügend Speicherplatz für Anwendungen und Betriebssystem verfügbar ist.
- Temperatur ??? Die ganzen Standardüberwachungen -> Thüllmann
- **Sessionanzahl** Anzahl der am CMS angemeldeten Benutzer, da aus Performanzgründen eine Obergrenze mit einer maximalen Anzahl festgelegt ist.

Prozesse

- **IdcServerNT.exe** Der Windowsprozess des Stellent-Servers
- **IdcAdminNT.exe** Der Windowsprozess für die Administration (Webinterface?) des Stellent-Servers

- **w3wp.exe** Der Windowsprozess des Microsoft „Internet Information Services“

Services

- **IdcContentService???** Den Zustand des „Content-Dienstes“-„`sccdms01`“ überprüfen.
- **IdcAdminService???** Den Zustand des „Administrations-Dienstes“-„`sccdms01_admin`“ überprüfen.
- **Zeitsynchronisationsdienst:** Überprüfen, ob der „`W32TIME`“-Dienst, der für den Zeitabgleich mit einem Zeitserver zuständig ist, läuft und die Abweichung zwischen Client und Zeitserver festhalten.
- **Antivirusdienst**: Den Zustand den Dienstes überprüfen, der für die ständig Updates des Virusscanners Symantec AntiVirus notwendig ist.

7.2 Überwachung der Funktionalität

Durch die vorherigen Tests kann herausgefunden werden, ob eine Anwendung oder ein Dienst auf dem Server gestarten wurde. Die Funktionalität kann durch solche Überprüfungen jedoch **nicht** sichergestellt werden. Da beispielsweise der Prozess bzw. Dienst des Webservers gestartet ist, jedoch keine Webseite aufgerufen werden kann. Daher muss eine weitere Art von Überprüfungen/Checks die Anwendungen auf ihre Funktionalität (hin) überprüfen.

- **Webserver** Aufruf einer Webseite auf dem Server. Wenn auf diese Anfrage eine gültige Antwort in Form einer Statuscode-Meldung erfolgt, kann der reale/wirkliche Zustand des Webservers festgestellt werden.
- **Webinterface des Oracle UCM** Zusätzlich wird mit dieser Abfrage die Integration des Conten-Management-Systems in den Webserver überwacht, da hier nicht nur der Webserver, sondern eine UCM spezifische Webseite abgefragt wird.

- **Benutzeranmeldung am Oracle UCM** Hier wird getesten, ob sich ein Benutzer erfolgreich am System anmelden kann. Dies wird mit Anmeldungsdaten eines lokalen Benutzers und eines Active Directory-Benutzers durchgeführt um gleichzeitig/zusätzlich die Verbindung zum ADS-Server zu testen.

- **Oracle Datenbank** Wenn keine Verbindung zur Oracle Datenbank möglich ist, können keine neuen Informationen gespeichert werden. (zitat riester: läuft das system noch pseudomäßig, wirft aber jede Menge Fehler)

- **Status von Cronjobs** In periodischen Zeitabständen werden Programme aufgerufen, deren Aufruf und Endstatus/Endergebniss überwacht werden muss. Damit nicht das vorherige Ergebnis zu einem False Negative führt, müssen hier zusätzliche Zeitinformationen/zeitliche Parameter beachtet/bedacht werden.

7.3 Auswerten von Logdateien

In dieser Kategorie werden zusätzlich verschiedene Logdateien auf spezielle Warnungs- und Fehlermeldungen mit eindeutigen Stopwörtern untersucht. Dies ist notwendig um reaktiv Fehlverhalten der Anwendung zu erkennen, das nicht mit den vorherigen Überwachungselementen entdeckt wurde. Desweiteren können durch die Analyse der Logdateien etwaige Alarmmeldungen der bisherigen Tests bestätigt, begründet oder aufgehoben werden. Somit bietet das Auswerten der Logdateien zusätzliche Sicherheit False Positive- oder False Negative-Meldungen auszuschliessen.
Die Oracle UCM Anwendung erstellt drei verschiedene Arten von Logdateien:[27]

- **Content Server Log**

- **Inbound Refinery Log**

[27] Quelle: [UCMlog09]

- **Archiver Log**

Um alle Logs ohne Probleme im Internetbrowser anzuzeigen, liegen alle Logdateien im HTML-Format vor. Alle drei Arten von Logs bestehen jeweils aus 30 verschiedenen Dateien, die sich täglich abwechseln. Dadurch wird für jeden Tag im Monat eine separate Datei verwendet, um bei vielen Warnungs- und Fehlermeldungen durch die chronologische Anordnung/Hierarchie den Überblick zu behalten. Dabei werden die Logdateien zwangsweise nach 30 Tagen nacheinander überschrieben.
Diese Rotation der Logdateien muss bei der Durchsuchung nach Signal/Stopwörter beachtet werden, damit stets die aktuelle Logdatei überwacht wird und keine veralteten Informationen für False Positives-Meldungen durch Nagios sorgen.

7.4 Benutzersimulation

- **Einchecken von Dokumenten** Damit die eigentliche Aufgabe des Dokumentenverwaltungssystem überwacht werden kann, werden verschiedene Datenformate testweise eingecheckt. Dabei wird die Antwort der Anwendung auf das Hinzufügen der Dateien analysiert.

- **Konvertierung** Da das hinzugefügte Dokument nicht nur einfach auf dem Server gespeichert wird, sondern dabei auch in ein anderes Format umgewandelt wird, muss diese Konvertierung zusätzlich überwacht werden. Wird beispielsweise ein Bild eingecheckt, wird dieses mehrfach in verschiedenen Auflösung oder als anderes Bilddateiformat gespeichert. Ob diese Transformation erfolgreich ablief, kann anhand dieser neuen Dateien festgestellt werden.

- **Indizierung** Bei dem Einchecken sollen auch gleichzeitig zusätzliche Informationen über das Dokument festgehalten werden. Diese Informationen können beispielsweise der Name des Authors, das Erstellungsdatum der Datei oder - bei Bildern - der verwendete Farbraum sein. Bei

der Suche nach einem Dokument können diese Informationen als zusätzliche Suchkriterien verwendet werden. Daher müss überprüft werden, ob diese Daten richtig ausgelesen werden, der Datenbank hinzugefügt und vom Anwender abgefragt werden können. Dabei werden auch zuvor festgelegte/ausgewählte Testdateien verwendet.

- **Suchfunktion** Nach einer erfolgreichen Indizierung muss das eingecheckte Dokument per Suchanfrage gefunden werden. Ob die Suche und Indizierung erfolgreich abgelaufen ist, wird zusätzlich überprüft.

Alle Überwachungselemente lassen sich inklusive ihrer Abhängigkeiten in einer Pyramide darstellen.

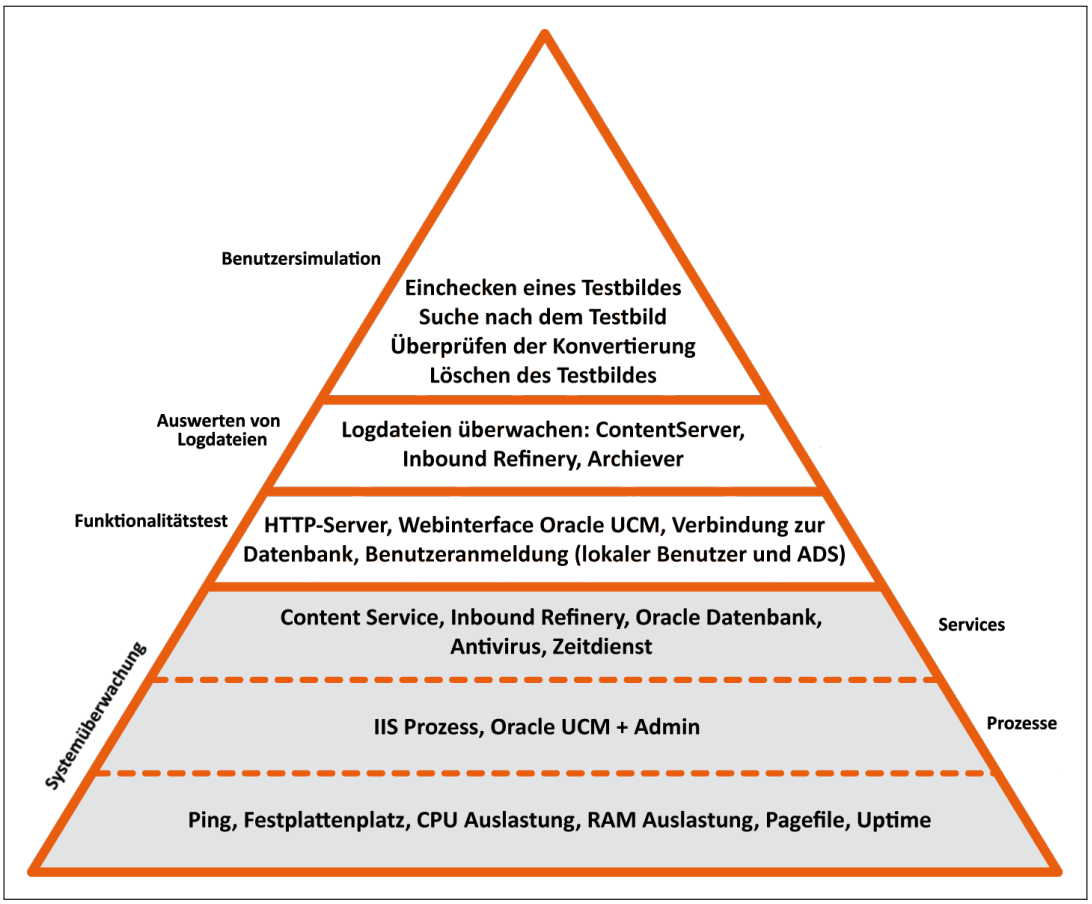


Abbildung 22: Überwachungselemente

Die Basis für die alle anderen Tests bildet die Systemüberwachung.

An erster Stelle der Systemüberwachung steht die schlichte Erreichbarkeit über das Netzwerk per Ping. Wenn der Server nicht erreichbar ist, können auch keine weiterführende Prüfungen durchgeführt werden. Zur Systemüberwachung gehören auch allgemeine Informationen über die Systemressourcen wie freier Festplattenspeicher oder CPU Auslastung.
Die nächste Grundlage für die darüberliegenden Tests bildet die Überprüfung der laufenden Prozesse und der Status verschiedener Dienste bzw. Services. Sollten bestimmte Prozesse nicht gefunden werden oder wichtige Dienste nicht gestartet sein, können auf diese Prozesse und Dienste aufbauende Checks nicht funktionieren. Beispielweise kann der Funktionalitätstest der Benutzeranmeldung nicht realisierbar sein, wenn bereits zuvor in der Systemüberwachung der Prozess für den Webserver IIS nicht gefunden werden konnte.

8 Umsetzung

In diesem Kapitel wird die Vorgehensweise der zuvor beschriebenen Problemstellungen erörtert.

8.1 Aufbau der Testumgebung

8.1.1 Aufsetzen eines Nagios-Test-Systems

Da die einzelnen Überwachungselemente in der Überwachungssoftware Nagios nach und nach eingetragen (/ definiert / assoziiert / verbunden) werden müssen, ist ein häufiges Neustarten der Nagios Anwendung notwendig, damit die neuen Konfigurationsdateien übernommen werden.
Damit dies nicht auf dem produktiven Nagios-Server durchgeführt werden muss, wird ein Nagios-Testserver für diesen Zwecks eingesetzt.

8.1.2 Bilddatenbank als VM

Für die Simulation der verschiedenen Fehlerzuständen der einzelnen Überwachungselemente wird eine virtuelle Maschine mit einer Oracle UCM Prototypinstallation, die extra als Entwicklungsplattform erstellt wurde, verwendet.

8.2 Einrichten des Windows Agenten

- Port ändern -> RPC
- Verschlüsselung durch PW und/oder Algo
- Hinzufügen der Plugins
- Bsp Aufruf aktiver Check

.Net 2.0 Framework essentiell NC_Net installieren nagios server ip zur sicherheit angeben port ändern pw hinzufügen -> dienst starten

test vom nagios host:

```
1 root@iwrpaul:/usr/local/nagios/libexec# ./check_nc_net -H secret.kit.edu -p
      123456 -s secret -v RUNSCRIPT -l check_uname.exe
2 Operating System OK - Microsoft(R) Windows(R) Server 2003 Standard Edition
      Service Pack 2
```

Listing 6: Aufruf eines aktiven Checks

Das auf dem Nagios Server liegende Script „`check_nc_net`“ stellt eine Verbindung zum angegebenen Server her und führt die mit dem Parameter „`l`“ angegebene Datei aus. Dafür muss sich diese Datei in dem Script Verzeichnis des NC_Net befinden.

Danach command definition hinzufügen, weil PW und Port verändert wurde:

```
1 # 'check_nt_bdb' command definition
2 #       _NSCLIENT_PORT  13599
3 #       _NSCLIENT_PW    KAnqloaQk
4 #
5 define command{
6     command_name    check_nc_net_bdb
7         command_line    /usr/lib/nagios/plugins/check_nc_net -H $HOSTNAME$ -
              p 13599 -s KAnqloaQk -v $ARG1$
8         }
```

Listing 7: Nagios-Befehls Definition für den Host

Danach

- Logfiles check.exe
- batchloader.exe script

8.3 Überprüfen der Prozesse und Services

- Prozesse
- Services
- Bsp Aufruf

8.4 Umsetzung der Funktionlitätstest

- batchloader + .blo Dateien
- einloggen mit lokalem und ads benutzer

8.5 Auswertung der Logs + Stopwörterdefinition

- check_logfiles
- check_logfiles cfg file inkl. Rotation

8.6 Benutzersimulation

- `http://www.w3schools.com/soap/default.asp` Web Services und SOAP
- `http://www.w3schools.com/wsdl/wsdl_summary.asp` WSDL
- max attempts bei Search erhöhen, da Auslastung der InboundRef -> möglichst keine/geringe False Positives -> auf Ausblick verweisen, Rahmenbedingungen müssen im Feld in der Praxis erst noch gefunden werden

ron

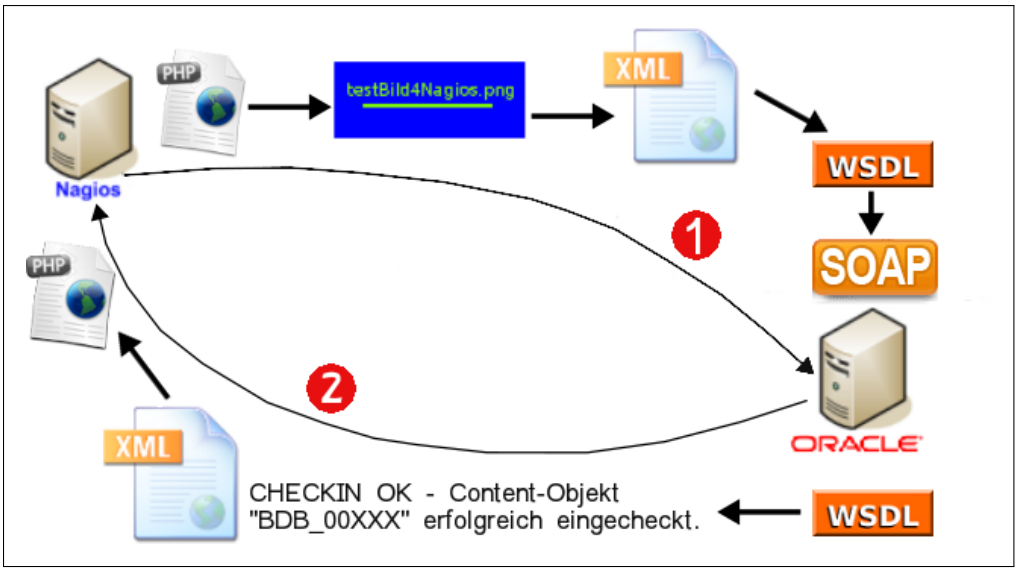


Abbildung 23: Ablauf der Benutzersimulation

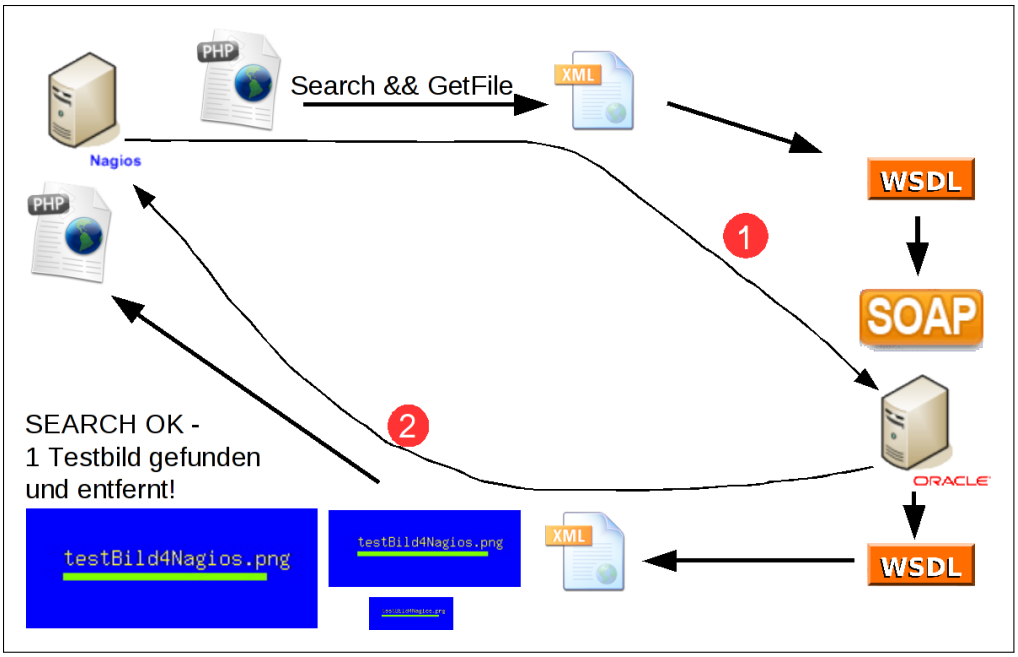


Abbildung 24: Überprüfung der Benutzersimulation

9 Ergebnis

- Vllt. Übersicht wie was überwacht wird
- Screenshots von Nagios
- Exportierfähigkeit, was muss alles auf dem Live-Nagios Server gemacht werden

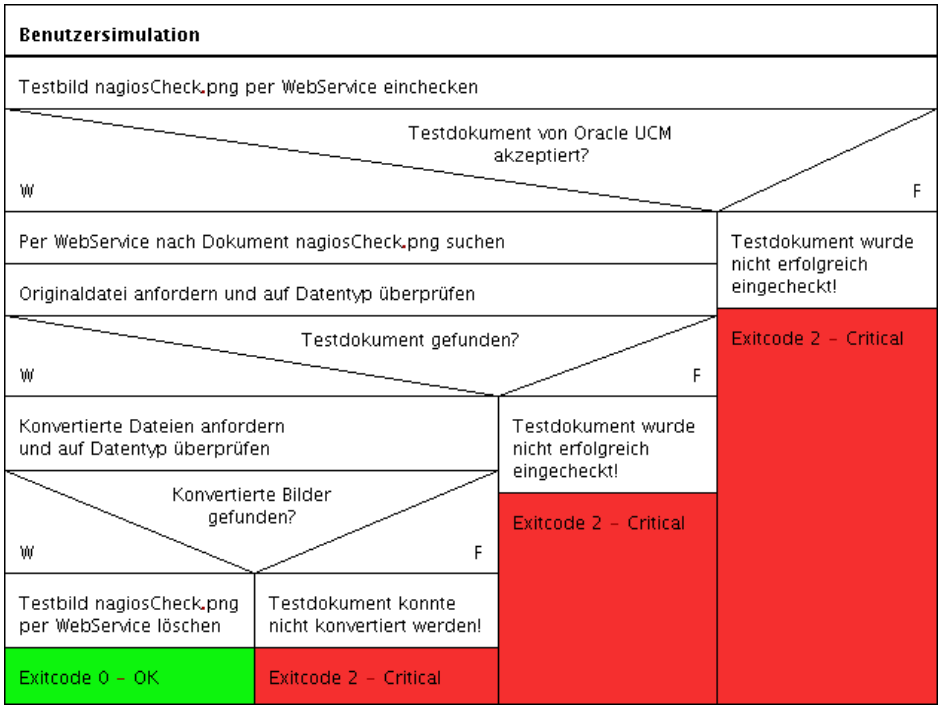
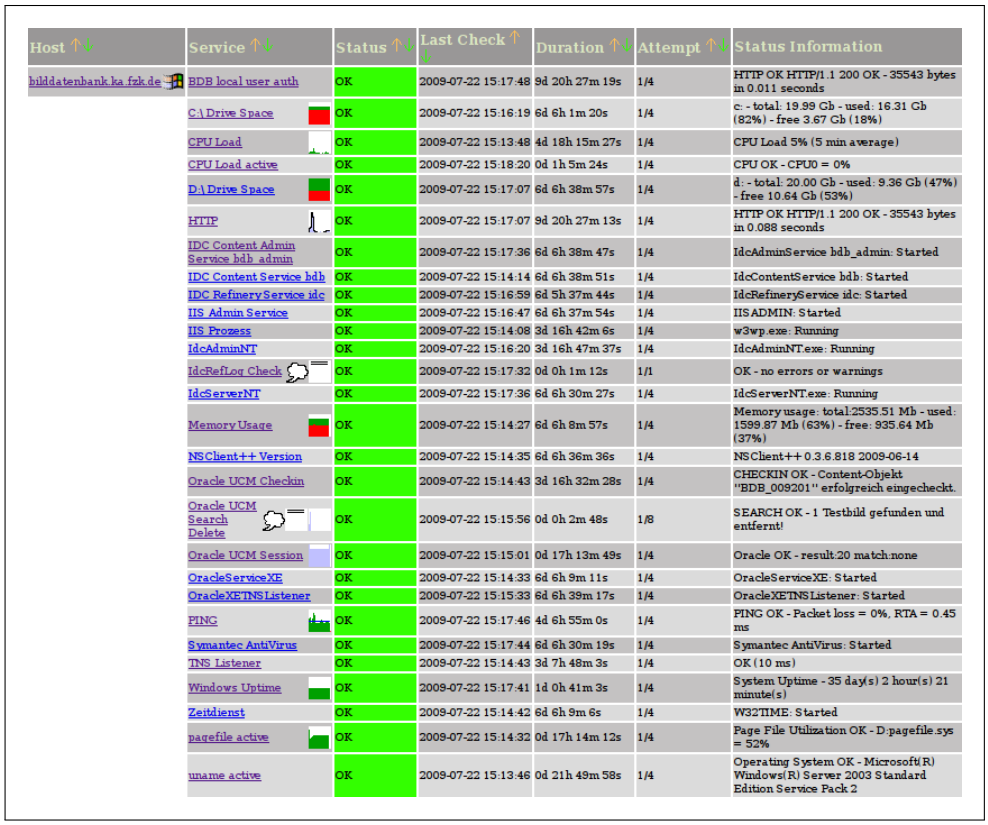


Abbildung 25: Ablauf der Benutzersimulation

10 Zusammenfassung und Ausblick

- Geeignete Stopwörter für Logdateien müssen noch gefunden / eruiert werden
- Passende Schwellwertdefinitionen können erst nach einer gewissen Laufzeit festegelegt werden
- Export der entwickelten Überwachung auf den produktiven Haupt-Nagios Server



Host	Service	Status	Last Check	Duration	Attempt	Status Information
bilddatenbank.ka.fzk.de	BDB local user auth	OK	2009-07-22 15:17:48	9d 20h 27m 19s	1/4	HTTP OK HTTP/1.1 200 OK - 35543 bytes in 0.011 seconds
	C:\ Drive Space	OK	2009-07-22 15:16:19	6d 6h 1m 20s	1/4	c: - total: 19.99 Gb - used: 16.31 Gb (82%) - free 3.67 Gb (18%)
	CPU Load	OK	2009-07-22 15:13:48	4d 18h 15m 27s	1/4	CPU Load 5% (5 min average)
	CPU Load active	OK	2009-07-22 15:18:20	0d 1h 5m 24s	1/4	CPU OK - CPU0 = 0%
	D:\ Drive Space	OK	2009-07-22 15:17:07	6d 6h 38m 57s	1/4	d: - total: 20.00 Gb - used: 9.36 Gb (47%) - free 10.64 Gb (53%)
	HTTP	OK	2009-07-22 15:17:07	9d 20h 27m 13s	1/4	HTTP OK HTTP/1.1 200 OK - 35543 bytes in 0.088 seconds
	IDC Content Admin Service bdb_admin	OK	2009-07-22 15:17:36	6d 6h 38m 47s	1/4	IdcAdminService bdb_admin: Started
	IDC Content Service bdb	OK	2009-07-22 15:14:14	6d 6h 38m 51s	1/4	IdcContentService bdb: Started
	IDC Refinery Service idc	OK	2009-07-22 15:16:59	6d 5h 37m 44s	1/4	IdcRefineryService idc: Started
	IIS Admin Service	OK	2009-07-22 15:16:47	6d 6h 37m 54s	1/4	IISADMIN: Started
	IIS Prozess	OK	2009-07-22 15:14:08	3d 16h 42m 6s	1/4	w3wp.exe: Running
	IdcAdminNT	OK	2009-07-22 15:16:20	3d 16h 47m 37s	1/4	IdcAdminNT.exe: Running
	IdcRefLog Check	OK	2009-07-22 15:17:32	0d 0h 1m 12s	1/1	OK - no errors or warnings
	IdcServerNT	OK	2009-07-22 15:17:36	6d 6h 30m 27s	1/4	IdcServerNT.exe: Running
	Memory Usage	OK	2009-07-22 15:14:27	6d 6h 8m 57s	1/4	Memory usage: total:2535.51 Mb - used: 1599.87 Mb (63%) - free: 935.64 Mb (37%)
	NSClient++ Version	OK	2009-07-22 15:14:35	6d 6h 36m 36s	1/4	NSClient++ 0.3.6.818 2009-06-14
	Oracle UCM Checkin	OK	2009-07-22 15:14:43	3d 16h 32m 28s	1/4	CHECKIN OK - Content-Objekt "BDB_009201" erfolgreich eingecheckt.
	Oracle UCM Search Delete	OK	2009-07-22 15:15:56	0d 0h 2m 48s	1/8	SEARCH OK - 1 Testbild gefunden und entfernt!
	Oracle UCM Session	OK	2009-07-22 15:15:01	0d 17h 13m 49s	1/4	Oracle OK - result:20 match:none
	OracleServiceXE	OK	2009-07-22 15:14:33	6d 6h 9m 11s	1/4	OracleServiceXE: Started
	OracleXETNSListener	OK	2009-07-22 15:15:33	6d 6h 39m 17s	1/4	OracleXETNSListener: Started
	PING	OK	2009-07-22 15:17:46	4d 6h 55m 0s	1/4	PING OK - Packet loss = 0%, RTA = 0.45 ms
	Symantec AntiVirus	OK	2009-07-22 15:17:44	6d 6h 30m 19s	1/4	Symantec AntiVirus: Started
	TNS Listener	OK	2009-07-22 15:14:43	3d 7h 48m 3s	1/4	OK (10 ms)
	Windows Uptime	OK	2009-07-22 15:17:41	1d 0h 41m 3s	1/4	System Uptime - 35 day(s) 2 hour(s) 21 minute(s)
	Zeitdienst	OK	2009-07-22 15:14:42	6d 6h 9m 6s	1/4	W32TIME: Started
	pagefile active	OK	2009-07-22 15:14:32	0d 17h 14m 12s	1/4	Page File Utilization OK - D:pagefile.sys = 52%
	uname active	OK	2009-07-22 15:13:46	0d 21h 49m 58s	1/4	Operating System OK - Microsoft(R) Windows(R) Server 2003 Standard Edition Service Pack 2

Abbildung 26: Webinterface von Nagios

11 Anhang

11.1 Abbildungsverzeichnis

11.2 Codelistingverzeichnis

11.3 Quellverzeichnis

11.3.1 Literaturverzeichnis

[DMS08] Götzer; Schmale; Maier; Komke (2008) „Dokumenten-Management - Informationen im Unternehmen effizient nutzen“ 4. Auflage,
dpunkt.verlag GmbH Heidelberg, ISBN13: 978-3-89864-529-4,
Stand: ????, Einsichtnahme: 25.06.2009

[Barth08] Wolfgang Barth (2008) „Nagios - System- und Netzwerk-Monitoring“ 2. Auflage,
ISBN13: 978-3-937514-46-8,
Stand: ????, Einsichtnahme: 25.05.2009

[Huff06] Brian Huff (2006) „The Definitive Guide to Stellent Content Server Development“,
ISBN13: 978-1-59059-684-5,
Stand: ????, Einsichtnahme: 25.05.2009

[Jose07] David Josephsen (2007) „Bulding a monitoring infrastructure with Nagios“,
ISBN13: 0-132-23693-1,
Stand: ????, Einsichtnahme: 16.06.2009

[SOA07] Hurwitz; Bloor; Baroudi; Kaufman; (2007) „Service Oriented Architecture For Dummies“,
ISBN13: 978-0-470-05435-2,
Stand: ????, Einsichtnahme: 29.07.2009

[Melzer08] Ingo Melzer (2007) „Service-orientierte Architekturen mit Web Services: Konzepte - Standards - Praxis“,
ISBN13: 978-9-8274-1993-4,
Stand: ????, Einsichtnahme: 29.07.2009

11.3.2 Internetquellen

[UCM07] Ohne Verfasser (2007) „Oracle Application Server Documentation Libary - Oracle Content Management 10gR3“,
Quelle: http://download-west.oracle.com/docs/cd/E10316_01/cs/cs_doc_10/documentation/integrator/getting_started_10en.pdf
Stand: unbekannt, Einsichtnahme: 16.06.2009

[UCMlog09] Unbekannter Author „vramanat“ (2009) „Universal Content Management 10gR3 - Content Server Log File Information“,
Quelle: http://www.oracle.com/technology/products/content-management/cdbs/loginfo.pdf
Stand: 20.01.2009, Einsichtnahme: 05.06.2009

[Munin08] Gabriele Pohl und Michael Renner (2008) „Munin - Graphisches Netzwerk- und System-Monitoring“,
ISBN13: 978-3-937514-48-2, Einsichtnahme: 05.04.2009

[OraPress] Letty Ledbetter (2009) „Oracle Press Release - Oracle Buys Stellent“,
Quelle: http://www.oracle.com/corporate/press/2006_nov/stellent.html
Stand: 02.11.2006, Einsichtnahme: 16.06.2009

[W3WS04] Booth; Haas; McCabe u.a. (2004) „Web Services Architecture - W3C Working Group Note 11 February 2004“,
Quelle: http://www.w3.org/TR/ws-arch/wsa.pdf
Stand: 11.02.2004, Einsichtnahme: 29.07.2009

11.4 Glossar

Bezeichnung	Beschreibung	Seiten
AIIM	Association for Information and Image	
Management	15	
CI	Coded Information	9, 11, 13
CMS	Content-Management-System	13–16
DMS	Dokumenten-Management-System	7, 13, 14
DNS	Domain Name System	31
DoS	Denial of Service	7
ECM	Enterprise-Content-Management	15
FTP	File Transfer Protokoll	27
HTTP	Hypertext Transfer Protocol	18
IP	Identifikation Protokoll	5, 7
LDAP	Lightweight Directory Access Protocol	31
MIB	Management Information Base	31
NCI	None-Coded Information	8, 9, 11, 13
NRPE	Nagios Remote Plugin Executor	29, 30

Bezeichnung	Beschreibung	Seiten
NSCA	Nagios Service Check Acceptor	32
OCR	Optical Character Recognition	11
OracleUCM	Content-Management-System von Oracle	44
PDF	Portable Document Format	11
SNMP	Simple Network Management Protocol	30–32
SOA	Service-Oriented Architecture	16–18, 36
SOAP	Simple Object Access Protocol	18–20
SSH	Secure Shell	29, 30
UCM	Universal-Content-Management	16
UDDI	Universal Description, Discovery and Integration protocol	19
W3C	World Wide Web Consortium	18
WSDL	Web Services Description Language	18–20
XML	Extensible Markup Language	18, 19