FERDINAND PORSCHE
FERN FH

Ablöse des Windows Monitoring-Agenten „NSClient++“ bei der ASFINAG IT

Bachelorarbeit

eingereicht von: **René Gau**
Matrikelnummer: 00230036

im Fachhochschul-Bachelorstudiengang Wirtschaftsinformatik (0470)
der Ferdinand Porsche FernFH

zur Erlangung des akademischen Grades eines

Bachelor of Arts in Business

Betreuung und Beurteilung: Christoph Jungbauer, BA MA MA

Wiener Neustadt, Juni 2024

Ehrenwörtliche Erklärung

Ich versichere hiermit,

1. dass ich die vorliegende Bachelorarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Alle Inhalte, die direkt oder indirekt aus fremden Quellen entnommen sind, sind durch entsprechende Quellenangaben gekennzeichnet.
2. dass ich diese Bachelorarbeit bisher weder im Inland noch im Ausland in irgendeiner Form als Prüfungsarbeit zur Beurteilung vorgelegt oder veröffentlicht habe.

Werndorf, 10.06.2024

Unterschrift

Creative Commons Lizenz

Die Kapitel 2 bis 4 der vorliegenden Bachelorarbeit wurden im Rahmen der Lehrveranstaltung „Bachelor Seminar 1“ eingereicht und am 02.02.2024 als Bachelorarbeit 1 angenommen.

Kurzzusammenfassung: Ablöse des Windows Monitoring-Agenten „NSClient++“ bei der ASFINAG IT

Da der derzeit bei der ASFINAG IT im Einsatz befindliche Windows Monitoring Agent NSClient++ nicht mehr weiterentwickelt wird, wird dafür eine Alternative zur Ablöse gesucht.

Mittels einer Nutzwertanalyse wird untersucht welches der sechs Windows Monitoring Agent Produkte (Centreon NSClient++, Nagios NRDP, tribe29 Checkmk, it-novum openITcockpit, Icinga, ConSol SNClient+) am besten geeignet ist, um den bisher verwendeten Windows Monitoring Agenten NSClient++ bei der ASFINAG IT abzulösen.

Es werden Muss- und Soll-Kriterien sowie Gewichtung dieser Kriterien festgelegt, die auf das bestehende ASFINAG IT Umfeld und dessen Bedürfnisse eingehen. Die sechs Alternativen werden anhand dieser Kriterien untersucht und ein Nutzwert für jede Alternative ermittelt. Die Alternative mit dem höchsten Nutzwert ist das Monitoring Agenten Produkt SNClient+.

Diese Alternative sollte einem Proof-of-Conecpt unterzogen werden, um die tatsächliche (technische) Realisierung zu überprüfen.

Schlagwörter:

NSClient++ SNClient+ Naemon Windows Monitoring Agent

Abstract: Replacement of the Windows monitoring agent "NSClient++" at ASFINAG IT

The Windows monitoring agent NSClient++ currently in use at ASFINAG IT is no longer being developed further and a replacement is required.

A utility analysis is used to examine which of the six Windows monitoring agent products (Centreon NSClient++, Nagios NRDP, tribe29 Checkmk, it-novum openITcockpit, Icinga, ConSol SNClient+) is best suited to replace the previously used Windows monitoring agent NSClient++ at ASFINAG IT.

Criteria’s as well as the weighting of these criteria are defined, which consider the existing ASFINAG IT environment and its needs. The six alternatives are examined based on these criteria and a utility value is determined for each alternative. The alternative with the highest utility is the monitoring agent product SNClient+.

The technical implementation of SNClient+ should be checked with a proof of concept.

Keywords:

NSClient++ SNClient+ Naemon Windows Monitoring Agent

Inhaltsverzeichnis

1. Einleitung

Die ASFINAG bzw. deren Auftragnehmer betreiben mehrere IT-Monitoringsysteme. Etwa ein eigenes Netzwerk-Monitoring/Management-System (NMS) für den Netzwerkbereich, ein eigenes Monitoringsystem, um Funktionalitäten von Web- oder Client-Anwendungen aus Benutzersicht zu überwachen (End-to-End Testing) oder spezielle Monitoring-Lösungen, um Container-Anwendungen zu überwachen. Weiters kommt ein klassisches System-Monitoring zum Einsatz. Dieses bestehende System-Monitoring überwacht unter anderem über einen Windows-Agenten etwa 3.000 Windows Systeme.

1.1 Ausgangslage

Für das System-Monitoring setzt die ASFIANG IT als Backend-System derzeit auf das open-source Projekt „Open Monitoring Distribution“ (OMD) in der Labs-Variante (OMD Labs Edition), welches federführend vom Unternehmen ConSol entwickelt wird [1]. Dies ist eine Zusammenstellung und Integration von mehreren open-source Tools welche sich historisch aus dem Nagios-Umfeld [2] herum entwickelt haben. Dabei kommt der Monitoring-Core „Naemon“ zum Einsatz [3].

Für die Überwachung von Windows-Systemen wird derzeit der Monitoring-Agent „NSClient++“[4] verwendet. Dabei fragen die Naemon Backend Systeme in regelmäßigen Intervallen die Windows Monitoring Agenten aktiv über das NRPE-Protokoll [5] ab (Polling).

1.2 Problemstellung

Das derzeit als Windows Monitoring Agenten eingesetzte Produkt „NSClient++” ist ein open-source Monitoring-Agent mit offenen Schnittstellen (wie das NRPE-Protokoll), welches von beliebigen Monitoring-Backendsystemen verwendet werden kann. Dieser Monitoring-Agent wird seit 2018 nicht mehr weiterentwickelt, ist jedoch beim Monitoring von Windows Umgebungen nach wie vor weit verbreitet.

Da eine nicht mehr aktiv weiterentwickelte Software-Lösung ein (Betriebs)Risiko darstellt (auftretende Schwachstellen werden nicht behoben, mit zukünftigen Betriebssystem-versionen nicht mehr kompatibel), ist eine Ablöse alternativlos.

1.3 Markterkundung

Für die Auswahl von Alternativen werden oftmals Markterhebungen herangezogen. Manche Marktforschungsunternehmen untersuchen von sich aus gewisse Märkte und veröffentlichen dazu dann eine Marktübersicht. Beispielsweise das

Marktforschungsunternehmen Gartner mit ihrer Marktuntersuchungsserie „Gartner's Magic Quadrant“ [6].

Für Windows Monitoring Agenten konnte allerdings keine vorhandene (aktuelle) Marktuntersuchung ausfindig gemacht werden. Daher hat die ASFINAG im Frühjahr 2023 die „CANCOM a+d IT solutions GmbH“ mit einer Markterkundung von alternativen Windows Monitoring Agenten - mit denen NSClient++ abgelöst werden kann - beauftragt.

Die Vorgaben zur Markterkundung waren rudimentär. Die Windows Monitoring Agenten Alternativen sollten (möglicherweise) dazu geeignet sein, den vorhandenen Windows Monitoring Agenten NSClient++ abzulösen, ohne dazu das vorhandene Monitoring Backend Naemon ablösen zu müssen.

Ziel der beauftragten Markterkundung war es, möglichst viele in Frage kommende alternative Windows Monitoring Agenten Produkte ausfindig zu machen. Somit eine aktuelle Übersicht über die Produktalternativen am Markt zu erhalten.

Durchgeführt wurde diese Markterkundung von der CANCOM mittels

- Internet-Recherche
- E-Mail-Anfragen an Hersteller bzw. Partnerunternehmen
- (Video)Telefonate mit Hersteller bzw. Partnerunternehmen

Folgende Windows Monitoring Agent Produkte wurden bei dieser Markterkundung von der CANCOM ausfindig gemacht:

- Centreon NSClient++ [7]
- Tribe29 Checkmk [8]
- Nagios NRDP [9]
- it-novum openITcockpit [10]
- Icinga [11]
- ConSol SNClient+ [12]

Die CANCOM sieht das Windows Monitoring Agent Produkt openITcockpit von der it-novum GmbH als am vielversprechendsten für die bestehende ASFINAG IT-Monitoring Umgebung an.

1.4 Ziel dieser Bachelorarbeit

Ziel dieser Bachelorarbeit ist es herauszufinden, mit welchem Windows Monitoring Agent Produkt das derzeit verwendete Produkt NSClient++ in der ASFINAG IT am besten abgelöst werden kann.

1.5 Forschungsfrage

Welches der sechs Windows Monitoring Agent Produkte (Centreon NSClient++, Nagios NRDP, tribe29 Checkmk, it-novum openITcockpit, Icinga, ConSol SNClient+) ist am

besten geeignet, um den bisher verwendeten Windows Monitoring Agenten NSClient++ bei der ASFINAG IT abzulösen?

1.6 Hypothese

Basierend auf die zuvor durchgeführte Markterkundung und Empfehlung wird folgende Hypothese aufgestellt:

Das Windows Monitoring Agent Produkt openITcockpit von der it-novum GmbH ist am besten geeignet um den bestehenden Windows Monitoring Agenten NSClient++ bei der ASFINAG IT abzulösen.

1.7 Methode

Mit den sechs in der zuvor durchgeführten Marktanalyse gefundenen alternativen Windows Monitoring Agent Produkten wird eine Nutzwertanalyse durchgeführt.

Die (technischen wie auch wirtschaftlichen) Kriterien, die Gewichtung der Kriterien (Bewertungsfaktoren) sowie die Bewertung (Zielerreichung) der einzelnen Kriterien werden genau spezifiziert (siehe Kapitel 4 Nutzwertanalyse).

Anschließend wir jede der sechs Alternativen anhand dieser Festlegungen bewertet und die Alternative mit dem höchsten Nutzwert bestimmt.

2. Arten von Monitoring Agenten und deren Funktionalitäten

In diesem Kapitel werden unterschiedliche Arten wie Monitoring betrieben werden kann beschrieben. Dabei wird darauf eingegangen nach welchen Prinzipien das bisher eingesetzte Produkt NSClient++ arbeitet und welche davon bei der ASFINAG derzeit zur Anwendung kommen.

2.1 Zustands-, Ereignis-, und Metrik-basiertes Monitoring

2.1.1 Zustandsbasiertes Monitoring

Die ASFINAG IT hat sich dazu entschieden, das Monitoring von u.a. der Windows Umgebung mittels eines zustandsbasierten Monitoringsystem aufzusetzen. Bei Zustandsveränderungen – etwa bei einer Abweichung von einem festgelegten Normalzustand – wird dies vom Monitoringsystem erkannt und ein Ereignis (Alarm) generiert, welches an das ITSM-System übermittelt wird. Das ITSM-System dient dabei als ereignisverarbeitendes System, über welches die erkannte Abweichung (Störung) entsprechend bearbeitet wird.

Bei einem Zustandsmonitoring werden Zustände ermittelt (abgefragt, gemessen…). Etwa ist das Licht an oder aus.

Ein Vorteil von zustandsbasierten Monitoringsystemen ist, dass jederzeit aktiv nach aktuellen Zuständen abgefragt werden kann und erkannt wird, wenn ein zu überwachendes Objekt auf Anfrage nicht antwortet.

2.1.2 Ereignisbasiertes Monitoring

Bei einem Ereignismonitoring werden Ereignisse ausgewertet. Beispielsweise Log-Daten oder Ereignismeldungen. Etwa das Licht ist ausgegangen. Bekannte Vertreter von ereignisbasierten Monitoringsysteme sind etwas Log-verarbeitende Systeme wie Splunk bzw. allgemein SIEM-Systeme.

In einer idealen Umgebung sind beide Varianten grundsätzlich gleichwertig nutzbar, wobei jede Variante ihre spezifischen Vor- und Nachteile hat.

Vorteile von ereignisbasierten Monitoringsystemen sind beispielsweise deren geringere Netzwerkbelastung (kein regelmäßiges Polling notwendig) und Zustandsänderungen unmittelbar verarbeitet werden können.

Ein Nachteil besteht darin, dass Ereignismeldungen verloren gehen können.

2.1.3 Zustands- als auch ereignisorientierte Systeme

Es gibt auch Technologien, die beide Varianten unterstützen. Bekannt dafür ist SNMP (Simple Network Management Protokoll). Dieses ist für zwei Arbeitsmodi spezifiziert. Mittels snmp-get kann ein Zustand abgefragt werden (z.B. die Temperatur in °C). Andererseits bietet SNMP die Möglichkeit, bei entsprechend hinterlegten Schwellwerten eine Ereignismeldung zu versenden (SNMP-Trap mit Meldung Temperatur wurde überschritten).

Der Windows Monitoring Agent NSClient++ wurde für die Verwendung in zustandsbasierten Monitoringsystemen entwickelt.

2.2 Aktives vs. passives Monitoring

Aus Sicht des Monitoring-Backend handelt es sich um ein aktives Monitoring, wenn das Backend-System ein zu überwachendes Objekt aktiv nach dessen Zustand abfragt.

Beispielsweise fragt ein Monitoring Backend-System bei einem Windows Monitoring Agenten nach, ob ein bestimmter Windows-Dienst auf Zustand „running“ ist. Der Monitoring-Agent ermittelt auf diese Anfrage hin den Zustand des gewünschten Windows-Dienstes und antwortet unmittelbar auf diese Anfrage.

Sollte ein Windows Monitoring Agent von sich aus (regelmäßig) den Zustand eines Windows-Dienstes ermitteln – ohne eine entsprechende Anfrage dazu erhalten zu haben - und das Zustandsergebnis an das Monitoring Backend-System gesendet wird, so spricht man (aus Sicht des Monitoring-Backend) von einem passiven Monitoring.

Bei ereignisbasiertem Monitoring handelt es sich immer um ein passives Monitoring. Zustandsbasiertes Monitoring kann sowohl aktiv wie auch passiv betrieben werden.

Agenten werden heutzutage oft für eine passive Betriebsart ausgelegt. Zwei Entwicklungen sind dafür ausschlaggebend:

- Informationssicherheit bzw. Cloud-Betriebsmöglichkeit
  Wenn sich das Monitoring-Backend in einer Internet-Cloud betrieben wird, so soll nicht aus der Cloud (bzw. aus dem Internet heraus) aktiv auf ein zu überwachendes System zugegriffen werden. Allgemein geht der Trend dahin, dass Agenten sich zu zentralen Backend-Systemen hin verbinden und nicht umgekehrt.

- Mikroservices bzw. Containertechnologien
  Die Anzahl der Systeme wird dynamisch an die Last angepasst. Es ist daher oft einfacher, dass die zu überwachenden Systeme (bzw. deren Agenten) ihre Monitoring-Daten an ein zentrales Monitoring-Backen übermitteln, als dass ein Monitoring-Backend sich laufend an die dynamischen Veränderungen „nachkonfiguriert“ und aktiv abfragt.
  Allgemein wird im Microservices-Umfeld sehr gerne mit Events als Kommunikationsstrategie gearbeitet. Beispielsweise Log-Events an ein zentrales Logmanagement-System gesendet.

Beim aktiven Monitoring liegt die Konfiguration des gesamten Monitorings primär beim (zentralen) Monitoring-Backend und ist über Parametrisierung der Abfragen sehr flexibel.

Beim passiven Monitoring liegt die Konfiguration des Monitorings primär außerhalb des Monitoring-Backend (etwa bei den Monitoring Agenten). Eine Synchronisation dieser verteilten Konfiguration zwischen Monitoring-Backend und Monitoring-Agenten kann daher sehr wichtig.

NSClient++ wurde primär für aktives Monitoring entwickelt. Ein Betrieb für passives Monitoring ist jedoch ebenfalls in NSClient++ eingebaut worden. Über das NSCA-Protokoll können Zustandsergebnisse an ein Nagios Monitoring-Backend übermittelt werden.

In der ASFINAG IT wird der Windows Monitoring Agent NSClient++ für aktives Monitoring eingesetzt.

2.3 Verteilte vs. zentrale Ermittlung von Zuständen

Als Beispiel soll die Disk-Auslastung eines Windows-Server dienen. Annahme ist, dass eine Diskauslastung bis zu 95% als Normalzustand gewertet werden soll (OK-Zustand) und ein darüber als Abweichung behandelt werden soll, welche näher zu untersuchen ist und daher entsprechend erkannt werden soll (CRITICAL-Zustand).

Im konkreten Fallbeispiel wird aktives Monitoring über einen Windows Monitoring Agenten angenommen.

Bei einer zentralen Ermittlung von Zuständen, wir die Zustandsermittlung durch den Monitoring-Core auf dem Monitoring Backend-System ermittelt.

Das Backend-System fragt beim Agenten nach der Festplattenauslastung und erhält von diesem beispielsweise die Rückmeldung 78%. Zusammen mit dem definierten Schwellwert von 95% kann der Monitoring-Core nun dein Zustand OK ermitteln.

Bei einer dezentralen Ermittlung von Zuständen, wir die Zustandsermittlung nicht vom Monitoring-Core durchgeführt, sondern von einer davor ausgelagerten Stelle. Die Schwellwertinformation wird dem Check-Plugin für die Zustandsermittlung mitübergeben. Das Monitoring-Backend fragt somit beim Windows Agenten nach dem Zustand der Diskauslastung unter Angabe der Schwellwertinformation an. Der Windows Monitoring Agenten meldet lediglich den Status zurück (in diesem Beispielfall OK). Die zugrundeliegenden Rohdaten (die aktuelle Disk-Auslastung) können dabei optional als sogenannte Performance-Daten an das Monitoring-Backend mitübermittelt werden.

Vorteil der zentralen Zustandsermittlung ist, dass die Konfiguration und Anpassungen von Zustandsauswertungen flexibler ist und zentral am Monitoring-Backend durchgeführt werden kann. Nachteil ist, dass immer alle Rohdaten übertragen werden müssen und die Rechenleistung der Auswertung zentral zur Verfügung gestellt werden muss.

Vorteil der dezentralen Zustandsermittlung ist, dass der Rechenaufwand dafür ebenfalls verteilt ist. Nachteil ist, dass die „Intelligenz“ der Zustandsauswertung bei den Monitoring Agenten liegt und bei Anpassungen der Auswertungsvorschrift (nicht nur parametrisierbarer Schwellwert) alle davon betroffenen Agenten entsprechend upgedatet werden müssen.

Das Produkt NSClient++ unterstützt lediglich den Ansatz der verteilten Zustandsermittlung. Wobei das als Monitoring Backend System eingesetzte Naemon ebenfalls nur eine verteilte Zustandsermittlung unterstützt.

2.4 Agentless Monitoring

Neben Agenten-basiertem Monitoring ist es auch möglich einen Monitoring Ansatz ohne Agenten zu betreiben.

Es ist ein großer Vorteil, dass keine Agenten verteilt, konfiguriert und upgedatet werden müssen. Nachteil ist, dass nur ein vorbestimmtes Set an Überwachungsmöglichkeiten bereitgestellt wird und keine oder nur sehr aufwendig individuellen Prüfmöglichkeiten implementiert werden können.

Agentless bedeutet in diesem Zusammenhang, dass auf einem System kein Dritt (Monitoring) Agent installiert werden muss, sondern das System selbst einen Monitoring Dienst zur Verfügung stellt, welcher abgerufen werden kann.

Die zu überwachende Systeme stellen einen standardisierten Dienst zur Verfügung, welcher von beliebigen Monitoring Backend Systemen genutzt werden kann. Beispiele dafür sind SNMP oder WMI.

Grundsätzlich ist es möglich die von Microsoft bereitgestellten Implementierungen von SNMP und WMI zu erweitern. Dies ist jedoch sehr aufwendig.

Die derzeit von Microsoft Windows zur Verfügung gestellten Dienste, welche zu Monitoringzwecken genutzt werden könnten, sind vom Leistungsumfang nicht ausreichend. Daher wird auch (zumindest in nächster Zeit) auch weiterhin ein eigener Windows Monitoring Agent bei der ASFINAG zum Einsatz kommen.

2.5 Metrikbasiertes Monitoring

Werden Messwerte in (regelmäßigen) Abständen abgespeichert, so entsteht eine sogenannte Zeitreihe (zeitlicher Verlauf einer Messgröße). Dies wird auch als Metrik bezeichnet.

Mittels Metriken lassen sich auch komplexere Abweichungsmuster erkennen.

Beispiel Disk-Auslastung:

- Zustandsbasiertes Monitoring
  Damit ist es möglich zu erkennen, ob die Disk voll ist oder z.B. ein Schwellwert von 90% erreicht ist.

- Metrikbasiertes Monitoring
  Die Möglichkeiten des Zustandsmonitorings sind auch beim metrikbasiertem Monitoring über Schwellwerte möglich.
  Jedoch ist es beispielsweise auch möglich die Steigung des Disk-Auslastungsverlaufs (der Metrik) auszuwerten. So ist es möglich zu erkennen, wenn statt durchschnittlich 10 MB pro Minute auf die Disk 10 GB pro Minute zusätzlich geschrieben werden.
  Über die erste Ableitung lässt sich die Steigung ermitteln und wenn diese einen gewissen Schwellwert überschreitet, so wird dies als Abweichung erkannt.

Ein weit verbreitetes metrikbasierendes Monitoringsystem ist etwa Prometheus.

Prometheus selbst besteht lediglich aus einer Monitoring Backend Komponente. Systemkomponenten, welche mittels Prometheus gemonitored werden sollen, müssen Metriken in einer für Prometheus lesbaren Form zur Verfügung stellen. Dies wird in der Prometheus Welt als Prometheus Exporter bezeichnet. Das Prometheus Backend sammelt regelmäßig von allen Exportern die bereitgestellten Metriken ab (polling).

Für Windows Systeme ist ein Prometheus Exporter mit der Bezeichnung windows_exporter verfügbar [13]

Dieser metrikbasierender Prometheus Windows Monitoring Agent kann jedoch nicht zusammen mit dem bestehenden zustandsbasierten Monitoring Backend Naemon betrieben werden. Daher stellt dieser Agent keine Alternative für die Ablöse von NSClient++ dar.

2.5.1 Von Metrik zu Zustand

Grundsätzlich möchte man von einem System wissen, ob es sich in einem normalen Zustand befindet oder eine Abweichung (Anomalie) aufweist.

Daher werden von einem metrikbasierten Monitoringsystem die Metriken regelmäßig ausgewertet und daraus ein (System)Zustand ermittelt.

Aus einer einzelnen Metrik lassen sich auch mehrere Zustände ermitteln. Auf das Disk-Auslastungsbeispiel bezogen etwa einmal ein Zustand, ob die absolute Diskauslastung sich in einem Normalzustand befindet und ein zweiter Zustand, ob sich die Diskauslastungsveränderung in einem Normalzustand befindet oder nicht.

Man muss den metrikbasierten Monitoringsystemen die Information geben (konfigurieren), wann eine Verlaufsform einer Metrik eine Abweichung darstellt. Dies wird meist als Alert Rule bezeichnet. Dies kommt historisch aus der ersten Anwendung dazu – bei einer erkannten Abweichung zu alarmieren.

Über die regelmäßige Auswertung von Metrikverläufen anhand von Auswertungsvorschriften (Alert Rules) entstehen somit Zustände.

Metrikbasierende Monitoringsysteme sind daher im Regelfall Monitoringsysteme mit zentraler Ermittlung von Zuständen (siehe Kapitel 2.3 Verteilte vs. zentrale Ermittlung von Zuständen).

2.5.2 Kompatibilität zwischen metrik- und zustandsbasierenden Monitoringsystemen

Inzwischen stellen metrikbasierte Monitoringsysteme im Regelfall diese Zustände über Schnittstellen zur Verfügung und dadurch können zustandsbasierte Monitoring Backend Systeme an metrikbasierende Monitoring Backend Systeme angebunden werden.

Bei Monitoring Backend Systemen, welche eine zentrale Ermittlung von Zuständen durchführen, ist die Anbindung an einen metrikbasierenden Agenten möglich. Beispielsweise wird dies von Checkmk unterstützt.

Jedoch können metrikbasierte Monitoring Agenten wie der zuvor genannte (Prometheus) windows_exporter nicht mit verteilt auswertenden metrikbasierten Monitoring Backend Systemen (wie naemon) verwendet werden, da die Auswertung der Metriken und damit die Zustandsgenerierung nicht auf Agentenseite erfolgt, sondern erst im Backend von metrikbasierenden Monitoringsystemen stattfindet.

Somit ist der Prometheus Windows Monitoring Agent windows_exporter keine mögliche Alternative für die Ablöse von NSClient++.

3. Aktueller Forschungsstand und bisherige Arbeiten

Zu Monitoring im Allgemeinen als auch IT-Monitoring gibt grundsätzlich es sehr viel Literatur.

3.1 Auswahl eines System- bzw. Netzwerkmonitoringsystems

Es gibt eine Vielzahl an Bachelor-, Master- und Diplomarbeiten, die sich mit der Auswahl eines geeigneten Monitoringsystems für ein konkretes Umfeld bzw. Organisationen befassen.

Beispielsweise eine Diplomarbeit von Peter Donko, die sich mit der Auswahl eines geeigneten Monitoringsystems für eine Tageszeitung beschäftigt [14]. Darin werden u.a. die Unterschiede von aktivem und passivem Monitoring näher beschrieben sowie die Vor- und Nachteile näher ausgeführt [14, S. 13–15].

Weiters werden darin auch der Aufbau von SNMP näher ausgeführt und auf die Unterschiede zwischen aktiver Abfrage (SNMP-Get) sowie (SNMP)Traps (passives Empfangen von Events) eingegangen [14, S. 23–28].

In der Diplomarbeit von Holm Schwantner werden einige Monitoring Systeme behandelt, die auch in dieser Arbeit und/oder im bestehenden ASFINAG Umfeld vorkommen [15]. Unter anderem werden Naemon als Nagios-Fork, openITcockpit, Checkmk und Icinga in dieser Diplomarbeit betrachtet [15, S. 44–45].

Allerdings geht es auch in dieser Diplomarbeit um die Auswahl eines Gesamtprodukts bzw. mehr um die Auswahl eines geeigneten Monitoring Backend System als um (Windows) Agenten.

3.2 Monitoring Lösungen bringen eigene Windows Agenten mit

Monitoring Lösungen bringen im Regelfall ihren jeweils eigenen Windows Monitoring Agenten mit. Wie etwa Zabbix [16] oder Checkmk.

Auch die meisten der von Nagios abstammenden Monitoringlösungen, welche ursprünglich keinen eigenen Windows Monitoring Agenten hatten, haben sich fast alle dazu entschlossen, jeweils eigene Windows Monitoring Agenten zu entwickeln. Wie beispielsweise Icinga oder openITcockpit.

3.2.1 Vendor Lock-in Effekt

In der Dissertation von Mathias Mormul, in der es um Konzepte zur Verbesserung des Monitoring in industriellen Cloud- Umgebungen geht, wird der Vendor Lock-in Effekt *„als ein zeit- und kostenintensiver Austausch eines Monitoring-Systems und dazugehöriger Agenten charakterisiert“* [17, S. 24]

Wenn Teilkomponenten eines Systems – wie etwa Monitoring Backend und Monitoring Agenten – über miteinander kompatible Schnittstellen verfügen, so ist es möglich, diese Teilkomponenten auszutauschen. Ansonsten muss bei einem Wechsel mit einem deutlich höheren Aufwand gerechnet werden.

Nachdem der Windows Monitoring Agent NSClient++ seit längerer Zeit nicht mehr weiterentwickelt und gepflegt wird, haben die Monitoring Backend Hersteller ohne eigenen Windows Monitoring Agenten sich dazu entschlossen, jeweils eigene Lösungen zu entwickeln.

3.2.2 Auseinanderentwicklung von Monitoringsystemen mit Nagios Ursprung

Das Ökosystem rund um Nagios war geprägt von Interkompatibilität. Bei Nagios Forks wie Icinga und Naemon konnten die gleichen Check-Plugins verwendet werden und auch der Windows Monitoring Agent NSClient++ konnte mit Icinga als auch Naemon als Backend verwendet werden. Der Check_MK (Windows) Agent konnte ursprünglich mit einer Nagios-Erweiterung betrieben werden.

Die Vorstellungen der Projekte und Unternehmen dahinter über die zukünftige Entwicklung gingen jedoch teilweise sehr weit auseinander und die Interkompatibilität macht es schwierig Weiterentwicklungen einfließen zu lassen.

Dies hat dazu geführt, dass sich die Projekte auseinanderentwickelt haben. Icinga hat beispielsweise die Konfigurationsstruktur auf Monitoring Backends völlig geändert und auch mehrere eigene Windows Monitoring Agenten entwickelt, die nur mit dem Icinga-Backend kommunizieren können

Aus einem Checkmk Agenten mit Nagios-Erweiterung ist ein völlig eigenständiges Monitoring Backend System entstanden. Der Checkmk Windows Agent ist nicht mehr mit einem Nagios Backendsystem nutzbar.

Die Interoperabilität zwischen Monitoring-Komponenten hat sehr unter dieser Entwicklung gelitten. Einzelne Module wie Monitoring Agent und Monitoring Backend sind nun nicht mehr beliebig miteinander austauschbar und verwendbar.

Auch andere Monitoring Hersteller/Projekte haben jahrelang kein Interesse an einer Interoperabilität gezeigt.

3.3 Herausforderungen bei klassischem Systemmonitoringansatz mit klassischen Agenten

Einige wissenschaftliche Arbeiten beschäftigen sich mit Unzulänglichkeiten, die ein klassischer System Monitoring Ansatz nicht erfassen kann.

3.3.1 Microservice Architekturen

In den letzten Jahren haben sich Microservice-Architekturen etabliert. Getrieben durch Containertechnologie-Infrastrukturen erfreuen sich diese Architekturen insbesondere für Cloud-Anwendungen immer größerer Beliebtheit.

Ein klassischer Monitoring-Ansatz, wie dieser z.B. beim Monitoring von Windows Umgebungen und deren Applikationen bisher zur Anwendung gekommen ist, kann für Microservices nicht praktikabel übertragen werden.

Somit haben sich andere Monitoring-Ansätze im Bereich der Microservice-Architekturen etabliert. Insbesondere das Metrik-basierte Monitoring mittels Prometheus [18].

Dario Urban untersucht in seiner veröffentliche Bachelorarbeit die Visualisierung von Mirocservice-Monitoring zu optimieren [19]. Darin werden Monitoring im Microserivce-Kontext als auch allgemein von verteilten Systemen behandelt [19, S. 6–7]. Weiters wird auf Prometheus als *„Tool für Sammlung von Metriken“* [19, S. 8] eingegangen.

3.3.2 Observability

Ein weiterer Kritikpunkt an klassischem Monitoring ist das Fehlen von Informationen über den inneren Zustand von Systemen.

Einem Blog von Dynatrace, welcher den Unterschied von Monitoring und Observability erklärt, ist zusammenfassend folgendes zu entnehmen [20]:

Wenn beispielsweise eine Metrik auf Veränderungen überwacht wird, die auf ein Problem hinweisen, dann ist das eine Überwachung (= Monitoring).
Wenn ein System jedoch zusätzliche Daten über seinen inneren Zustand aussendet, die für die Bestimmung der Grundursache eines Problems bedeutend sind, so wird ein System beobachtbar und man spricht dann von Observability.

Andrew Magusson ergänzt dazu in seinem Blog zum Unterschied zwischen Observability und Monitoring:

„Observability ist eine Lösung, die alle von allen IT-Systemen erzeugten Daten aggregiert.“ [21]

In der Literatur wird oftmals von den drei Säulen der Observability geschrieben [22, S. 10]:

- Metriken
- Logs
- Traces

Wobei immer wieder auch weitere Säulen genannt werden. Graf sieht beispielsweise in seiner Arbeit die Visualisierung als zusätzliche Säule der Observability [22, S. 10].

Metriken und Logs (Events) wurden bereits behandelt. Auf Traces wird hier nicht näher eingegangen.

Zusammenfassend geht es darum, dass es mehrere Arten von Informationen gibt, die Rückschlüsse auf den inneren Zustand von System ermöglichen. Durch die Verknüpfung dieser Daten wird eine Beobachtbarkeit ermöglicht. Dadurch wird eine mehrwertbringende Ergänzung zum klassischen Monitoring erreicht.

3.4 Bedarf an Interoperabilität

Verteilte Systeme wie Microservice-Architekturen sind mit klassischen Werkzeugen nur sehr aufwendig zu analysieren (debuggen) und zu monitoren. Als Antwort darauf hat sich der Observability-Ansatz entwickelt.

Monitoring-Tool Hersteller haben für jede zu überwachende Systemart eine eigene Anbindung oder Agenten entwickeln. Beispielsweise hat (fast) jeder Monitoring-Tool Hersteller einen eigenen Agenten für Windows Systeme entwickelt.

Bei Microservices müsste nun jeder Observability-Tool Hersteller für jedes (weiterverbreitete) Microservice eine eigene Anbindung entwickeln. Etwa für die weit verbreiteten Microservice Message Broker Redis, Kafka und RabbitMQ.

Weiters müsste jeder Software-Hersteller für seine selbst entwickelten Microservices eine Anbindung zum jeweiligen Observability-Tool implementieren. Analog zu individuellen Check-Plugins, um Windows Monitoring an individuelle Gegebenheiten zu erweitern.

Im Bereich von Metriken im Bereich der Microservices ist Prometheus sehr verbreitet. Beispielsweise bringt die Container-Plattform OpenShift Prometheus standardmäßig mit und Metriken der OpenShift Infrastrukturplattform werden in Prometheus bereitgestellt.

Weiters gibt es für die gängigen Microservice Middleware Produkte sogenannte Prometheus Exporters. Über diese werden Metriken für Prometheus zur Verfügung gestellt. So existieren auch solche Exporters für die zuvor erwähnten Tools Redis, Kafka und RabbitMQ.

Wie bereits in Kapitel 2.5 Metrikbasiertes Monitoring erwähnt, gibt es auch einen Prometheus Exporter für Windows Systeme (windows_exporter).

3.4.1 OpenMetrics

Mit dem OpenMetrics Projekt [23] gibt es Bestrebungen eine frei verfügbare Standardisierung für ein Protokoll zur Übermittlung von Metriken zu erreichen. Dieses ist stark vom von Prometheus verwendeten Protokoll beeinflusst.

Hersteller von Observability-Tools haben im Regelfall eine Schnittstelle zu Prometheus und können darin enthaltene Metriken übernehmen und analysieren. Es ist davon auszugehen, dass die Observability-Tool Hersteller auch OpenMetrics implementieren werden.

Wie in Kapitel 2.5.2 Kompatibilität zwischen metrik- und zustandsbasierenden Monitoringsystemen ausgeführt, können auch klassische zustandsbasierte Monitoring Systeme an metrikbasierende Monitoring Systeme angebunden werden. Somit profitieren auch klassische Monitoring Lösungen vom weit verbreiteten Quasi-Standard Prometheus bzw. OpenMetrics

Aber nicht nur Software (Anwendungen, Middleware-Komponenten wie Redis, Software-Infrastruktur wie OpenShift) können Metriken für Monitoring oder Observability bereitstellen. Auch für eine darunter liegende Virtualisierungsplattform wie VMware gibt es vom Hersteller bereitgestellte Prometheus Anbindungen. Open-Source Projekte bieten sogar für (Server)Hardware Prometheus Exporter an. Dabei werden z.B. für HP die iLO-Schnittstelle und für Dell die iDRAC-Schnittstelle verwendet. Es ist wohl nur eine Frage der Zeit bis Hardware-Hersteller selbst eine Prometheus oder OpenMetrics Implementierung bereitstellen.

3.4.2 OpenTelemetry

Das OpenTelemetry Projekt [24] hat es sich zum Ziel gesetzt, herstellerunabhängige Spezifikationen für Telemetriedaten wie Metriken, Logs und Traces zu erstellen. Sowie API’s, SDK’s und Tools für die Instrumentierung, Generierung, Sammlung und Export von Telemetriedaten (Metriken, Logs und Traces) bereit zu stellen.

OpenTelemetry geht somit über OpenMetric hinaus und hat einen gesamtheitlicheren Ansatz. Beide Projekte werden von der Cloud Native Computing Foundation [25]

betrieben, in der über 600 Hersteller und Betreiber vertreten sind (u.a. RedHat, VMware, Microsoft, Google, Amazon, Alibaba, SAP, Oracle, Grafana, Cisco, Splunk).

Klassische zustandsbasierende Monitoring Systeme werden die Möglichkeiten von OpenTelemetry nicht vollumfänglich nutzen können. Jedoch wäre eine parallele Nutzung von OpenTelemetry sowohl für Observability als auch Monitoring Backend Systeme (etwa Metriken) möglich.

Wenn jedes System eine OpenTelemetry Schnittstelle zur Verfügung stellt, so wären keine eigenen Implementierungen oder Agenten notwendig. Wenn beispielsweise Microsoft eine OpenTelemetry Schnittstelle in das Windows Betriebssystem integriert sowie individuelle Erweiterungen ermöglicht, dann wäre ein Agentless Monitoring bei Windows Systemen möglich.

3.5 Zusammenfassung des aktuellen Forschungsstandes

Die ASFINAG IT setzt mit Naemon ein Monitoring Backend System ein, für welches (bisher) kein eigener Windows Monitoring Agent entwickelt wurde. Obwohl es sehr viel Literatur zu Monitoring gibt, konnten keine wissenschaftlichen Arbeiten gefunden werden, die sich mit der Herausforderung einer Ablöse des Windows Monitoring Agenten NSClient++ beschäftigt.

Arbeiten zur Auswahl von Monitoring Systemen beschäftigen primär mit Monitoring Backend Systemen bzw. geeigneten Monitoring Strategien und nicht konkret mit Windows Monitoring Agenten.

Auch Arbeiten zu Mängeln an bestehenden Monitoring Strategien, Monitoring Backend Systemen und Monitoring Agenten können nicht in bei der konkreten Forschungsfrage weiterhelfen.

Der aktuelle Trend der Standardisierung (OpenMetrics bzw. OpenTelemetry) hilft auch nicht weiter, da solche Agentenimplementierungen nicht direkt mit Naemon als Monitoring Backend System kompatibel sind.

Andere Unternehmen und Organisation, welche ebenfalls die Kombination aus Naemon (oder ein artverwandtes Produkt) als Monitoring Backend System in Kombination mit NSClient++ als Windows Monitoring Agenten einsetzen, können daher von dieser Bachelorarbeit profitieren.

4. Nutzwertanalyse

Als wissenschaftliche Methode (siehe Kapitel 1.7) zur Beantwortung der Forschungsfrage (siehe Kapitel 1.5) kommt eine Nutzwertanalyse zum Einsatz.

4.1 Allgemeines zu Nutzwertanalyse

Es gibt sehr viel Literatur zu Nutzwertanalysen. Nutzwertanalysen werden häufig auch bei Bachelor- und Masterarbeiten als wissenschaftliche Methode eingesetzt, um eine Forschungsfrage zu untersuchen. Exemplarisches Beispiel ist die veröffentlichte Bachelorarbeit von Weimert, welche CRM-Systeme auf Eignung und Nutzen für Affiliate Marketing untersucht hat [26].

Eine Nutzwertanalyse *„ermittelt den in Zahlen ausgedrückten subjektiven Wert von Lösungen und/oder Maßnahmen in Bezug auf die Zielvorgaben“* [27, S. 141]. Es fließen somit subjektive Einschätzungen und Informationen in die Bewertung der Alternativen ein.

Die Nutzwertanalyse wird laut Winkelhofer in folgende vier Schritte gegliedert [27, S. 140]:

Formulierung der Kriterien

- Gewichtung der Kriterien
- Bewertung der Kriterien der einzelnen Lösungsalternativen
- Errechnung des Nutzwertes für die einzelnen Lösungsalternativen

Das Ziel einer Nutzwertanalyse ist die Reihung der verschiedenen Lösungsalternativen je nach Vorliebe des Entscheidungsträgers [28].

Nach der Errechnung der Nutzwerte für die einzelnen Lösungsalternativen kann somit abgelesen werden, welche Alternative den meisten Nutzen stiftet [26, S. 41]

Weitere wichtige Eckpunkte einer Nutzwertanalyse:

- Kriterien müssen Unabhängigkeit untereinander sein. Es dürfen zwischen Kriterien keine Abhängigkeiten bestehen.

- Bei der Gewichtung der Kriterien ist darauf zu achten, dass *„in der Summe die einzelnen Anteile zusammen 100% ergeben“* [27, S. 141].

- Um die Bewertungskriterien zu präzisieren sind die Kriterien in Muss-Kriterien (auch KO-Kriterien genannt) und Soll-Kriterien zu gliedern.

4.2 Formulierung der Kriterien

Die Kriterien orientieren sich an denen der Markterkundung. Wobei diese einem Review unterzogen werden. Beispielsweise sind nicht alle Kriterien der Markterkundung voneinander unabhängig. Vorhandene Abhängigkeiten werden aufgelöst.

4.2.1 Muss-Kriterien

Muss-Kriterien sind zwingend zu erfüllen. Bei der Nicht-Erfüllung eines Muss-Kriteriums scheidet die Alternative aus.

Muss-Kriterien müssen so formuliert sein, dass als Bewertung nur mit einem erfüllt oder nicht-erfüllt erfolgen kann. Nicht gestattet sind Formulierungen, die eine Teilerfüllung zulassen würden.

- Muss Kriterium 1: **Weiterentwicklung des Agenten**

  Der Windows Monitoring Agent muss aktuell weiterentwickelt werden.

  Insbesondere muss gegeben sein, dass sicherheitsrelevante Software-Schwachstellen behoben werden.

- Muss Kriterium 1: **Unterstützung individueller Checks**

  Der Windows Monitoring Agent muss die Möglichkeit bieten, dass dieser um individuelle Checks erweitert werden kann.

  Dies ist bei der Windows IT-/Applikationslandschaft der ASFINAG unbedingt erforderlich. Wenn vorhandene Sonderlösungen nicht mehr automatisiert überwacht werden könnten, so hätte dies eine signifikante negative Auswirkung auf den Mehrwert der bestehenden IT-Monitorings. Dies ist unbedingt zu verhindern.

4.2.2 Soll-Kriterien

Soll-Kriterien sind wünschenswert. Bei einer Teil- oder auch Nicht-Erfüllung wird die Alternative jedoch nicht sofort ausgeschlossen.

- Soll Kriterium 1: **Kompatibilität mit Naemon**

  Die Weiterverwendung des bestehenden Monitoring Backend System Naemon soll technisch möglich sein. Weiters soll möglichst wenig Anpassungsbedarf an der Naemon Backend Konfiguration notwendig sein.

- Soll Kriterium 2: **Weiterverwendung individueller Check-Plugins**

  Die ASFINAG hat mit Stand August 2023 etwa 125 individuelle Windows Check-Plugins im Einsatz, die den „Monitoring Plugins Development Guidelines“ entsprechen. Diese sind bis auf wenige Ausnahmen Microsoft PowerShell Scripts.

  Die Weiterwendung von dieser vorhanden Monitoring Check-Plugins, ist von großem Vorteil, da ansonsten alle bestehenden Check-Plugins angepasst werden müssen.

- Soll Kriterium 3: **Supportunterstützung**

  Es soll die Möglichkeit geben, Support für den Windows Monitoring Agenten zu erhalten.

  Bei Problemen mit dem Agenten soll es Unterstützungsmöglichkeiten geben. Bewertet werden soll der Umfang der angebotenen Supportleistungen.

- Soll Kriterium 4: **Kosten für Lizenzierung und Support**

  Die direkten (laufenden) Kosten für Lizenzierung und Support sollen bewertet werden.

- Soll Kriterium 5: **Verwaltung der Agenten**

  Integrierte Unterstützung des Produkts, um die Konfiguration des Windows Monitoring Agenten sowie die Verteilung von individuellen Check-Plugins zu den Agenten zu bewerkstelligen.

Die ASFINAG hat derzeit etwa 3.000 Windows Systemen in Einsatz, auf welchen der Windows Monitoring Agent im Einsatz ist und abgelöst werden soll.
Der derzeit eingesetzte Windows Monitoring Agent NSClient++ bietet keinerlei Unterstützung die Agenten (zentral) zu verwalten. Sowohl die Installation der Agenten-Software, das Updaten der Agenten-Software auf eine neue Version, die lokale Agenten-Konfiguration als auch die Verteilung der benötigten individuellen Check-Plugins müssen über Drittlösungen erfolgen.

Bei einem individuellen Monitoring-Erweiterungsbedarf muss daher im Regelfall derzeit sowohl das Monitoring-Backend als auch die Monitoring-Agent Konfiguration angepasst werden und am betroffenen Windows-Host das dazu notwendige individuelle Check-Plugin deployt werden.

Daher sind Möglichkeiten für die (zentrale) Verwaltung der Agenten, deren (mitunter individuellen) Konfiguration sowie der Verteilung von Check-Plugins sehr interessant, da dadurch laufender Aufwand und damit laufende Kosten eingespart werden können.

- Soll Kriterium 6: **Interkompatibilität**

  Um einem Vendor Lock-In vorzubeugen, soll die Windows Monitoring Agent Alternative auch zu weiteren Monitoring Backend Systemen kompatibel sein.

Anmerkungen:

Der in der Markterkundung enthaltene Bewertungspunkt „Grobschätzung einmaliger (Kosten)Aufwand für Migration“ wird nicht als Kriterium in der Nutzwertanalyse aufgenommen. Grund dafür ist, dass der Migrationsaufwand und die damit verbundenen Kosten im Wesentlichen von den Soll-Kriterien 1, 2 und 5 abhängig sind.

Manche Punkte aus der Markterkundung wurden zusammengefasst zu einem Soll-Kriterium der Nutzwertanalyse. Dies geschah aus dem Grund, damit die einzelnen Kriterien unabhängig sind.

4.3 Gewichtung

Die Soll-Kriterien müssen nun gewichtet werden.

Über die Markterkundung (Auswahl der Alternativen) sowie die Muss-Kriterien wird sichergestellt, dass nur geeignete Alternativen für eine Reihung übrigbleiben.

Über die Gewichtung der Soll-Kriterien kann nun gesteuert werden, dass (technische) Mehrwerte als auch betriebswirtschaftliche Überlegungen (Kosten-Nutzen) entsprechende Berücksichtigung finden.

Die Kostenaufwendungen können unterschieden werden zwischen:

- Einmaliger Umstellungsaufwand (Transition-Kosten)
- Laufende Aufwände für
    - Anpassungen und Betrieb
    - Laufende Lizenzierung und Support-Pauschalen

Die laufenden Aufwände für Anpassungen und Betrieb werden grundsätzlich lösungsunabhängig auf etwa die gleiche Größenordnung geschätzt. Gewisse Kostenvorteile können sich aus Soll-Kriterium 6 „Verwaltung der Agenten“ geben.

Die großen Kostenblöcke sind bei den Transition-Kosten sowie bei laufenden Lizenzierungs- und Supportkosten zu erwarten. Wobei ein besonderes Augenmerk auf die Transition-Aufwände gelegt wird.

Je einfacher die Ablöse sich technisch darstellt, umso weniger Aufwände (Kosten) entstehen, umso schneller kann die Ablöse durchgeführt werden und umso weniger (notwendige) Änderungen wirken sich auf Anwender und Prozesse aus.

Basierend auf diesen Überlegungen werden die in „Tabelle 1: Gewichtungsfaktoren“ ersichtlichen Bewertungsfaktoren für die Soll-Kriterien festgelegt.

Soll Kriterium	**Kurzbezeichnung**	**Bewertungsfaktor**
1	Kompatibilität mit Naemon	0,25
2	Weiterverwendung individueller Check-Plugins	0,25
3	Supportunterstützung	0,15
4	Kosten für Lizenzierung und Support	0,25
5	Verwaltung der Agenten	0,05
6	Interkompatibilität	0,05
	Gesamt	**1,00**

Tabelle 1: Gewichtungsfaktoren

Anmerkung:

Wie einleitend in Kapitel 4 festgehalten, ist eine Nutzwertanalyse eine subjektive Bewertungsmethode. U.a. wegen der subjektiven Festlegung der Bewertungskriterien

4.4 Bewertung der Muss-Kriterien

4.4.1 Bewertung Muss-Kriterium 1: Weiterentwicklung des Agenten

Als Bewertung wird herangezogen, dass in den letzten 12 Monaten zumindest zwei Aktualisierungen der Windows Monitoring Agent Software veröffentlicht wurden. Weiters darf keine Abkündigung oder dergleichen veröffentlich worden sein.

Diese Bewertung für dieses Kriterium ist im Vergleich zur Markterkundung verschärft worden. Damit soll verhindert werden, dass eine Alternative eingesetzt wird, welche zeitnah wiederum abgelöst werden muss.

4.4.2 Bewertung Muss-Kriterium 1: Unterstützung individueller Checks

Der Windows Monitoring Agent muss in irgendeiner Form unterstützen, dass individuelle definierte Monitoring-Checks ermöglicht werden.

Im Regelfall haben Monitoring Agenten eine Möglichkeit über einen Plugin-Mechanismus die Check-Funktionalitäten zu erweitern.

4.5 Bewertung der Soll-Kriterien

Für jedes Kriterium werden jeweils eigenständige Bewertungsvorschriften festgelegt. Die Alternativen werden zu jedem Kriterium untersucht und anhand der jeweiligen Bewertungsvorschrift wird jeder Alternative ein Zielerfüllungsfaktor zugewiesen.

„Der Grad der Zielerfüllung wird durch den Zielerfüllungsfaktor ausgedrückt“ [28, S. 98]

Der Zielerfüllungsfaktor stellt eine Punktezahl dar. Alle Soll-Kriterien zielen auf denselben Höchstwert ab.

Mit der Anzahl an zu erreichenden Punkten wird eine Bewertungsgranularität gesteuert. Für diese Arbeit ist eine vierstufige Skala mit Punktezahl 0 bis 3 ausreichend granular.

4.5.1 Bewertung Soll-Kriterium 1: Kompatibilität mit naemon

- Zielerfüllungsfaktor 0
  Der Windows Monitoring Agent ist nicht mit dem Monitoring Backend naemon kompatibel.
  Beim Einsatz dieser Alternative ist ein Wechsel der Monitoring Backend Umgebung notwendig.

- Zielerfüllungsfaktor 1
  Der Windows Monitoring Agent ist zumindest teilweise mit dem Monitoring Backend naemon kompatibel.
  Am Monitoring Backen müssen aufwendige Anpassungen der Backend-Konfiguration und/oder Erweiterungen am Backen vorgenommen werden.
  Dennoch müssen teilweise Einschränkungen in Kauf genommen werden.

- Zielerfüllungsfaktor 2
  Der Windows Monitoring Agent ist weitgehend mit dem Monitoring Backend naemon kompatibel.
  Es müssen einige Anpassungen am Monitoring Backend und dessen Konfiguration vorgenommen werden.
  Es gibt keine oder kaum funktionelle Einschränkungen.

- Zielerfüllungsfaktor 3
  Der Windows Monitoring Agent ist voll mit dem Monitoring Backend naemon kompatibel.
  Es müssen keine oder nur geringe Anpassungen an Monitoring Backend Konfiguration vorgenommen werden

4.5.2 Bewertung Soll-Kriterium 2: Weiterverwendung individueller Check-Plugins

- Zielerfüllungsfaktor 0
  Die bisher entwickelten Powershell-Scripts können gar nicht weiter verwendet werden. Eine völlige Neuimplementierung aller Check-Plugins (etwa in einer anderen Programmiersprache) ist notwendig.

- Zielerfüllungsfaktor 1
  Von den bisher entwickelten Powershell-Scripts kann zumindest teilweise Code übernommen werden. Jedoch sind aufwendige Anpassungen notwendig.

- Zielerfüllungsfaktor 2
  Die bisher entwickelten Powershell-Scripts können weitgehend übernommen werden. Es sind lediglich kleinere Anpassungen notwendig.

- Zielerfüllungsfaktor 3
  Die bisher entwickelten Powershell-Scripts können eins-zu-eins ohne irgendwelche Anpassungen übernommen werden.

4.5.3 Bewertung Soll-Kriterium 3: Supportunterstützung

- Zielerfüllungsfaktor 0
  Es wird keinerlei (kommerzieller) Support für den Windows Monitoring Agenten Software angeboten. Weder vom Hersteller noch von anderen Marktteilnehmern.
  Beispielsweise bei einer open-source Lösung für welche keinerlei kommerzieller Support angeboten wird.

- Zielerfüllungsfaktor 1
  Es werden rudimentäre (kommerzielle) Supportunterstützungen rund um den Windows Monitoring Agenten angeboten. Jedoch lediglich etwa die Handhabung und Konfiguration. Jedoch keine Angebote, die Agentensoftware (gegen Beauftragung) auftretende Bugs zu fixen oder bei Bekanntwerden von Schwachstellen diese zu beheben.

- Zielerfüllungsfaktor 2
  Es werden (kommerzielle) Supportleistungen zumindest auf best-effort Basis angeboten, die auch die Agentensoftware umfassen.

- Zielerfüllungsfaktor 3
  Es werden umfangreiche kommerzielle Supportleistungen samt SLA für den Windows Monitoring Agenten angeboten.
  Beispielsweise gibt es eine Zusicherung, dass bei Bekanntwerden von kritischen Schwachstellen, diese innerhalb einer gewissen Zeit behoben werden und eine aktualisierte Agenten-Version bereitgestellt wird.

4.5.4 Bewertung Soll-Kriterium 4: Kosten für Lizenzierung und Support

- Zielerfüllungsfaktor 0
  Es fallen sehr hohen Lizenz- und Supportkosten in Höhe von mehr als € 100.000,- pro Jahr an.

- Zielerfüllungsfaktor 1
  Es fallen hohe Lizenz- und Supportkosten in Höhe zwischen € 50.000,- und € 100.000,- pro Jahr an.

- Zielerfüllungsfaktor 2
  Es fallen mittelhohe Lizenz- und Supportkosten in Höhe zwischen € 10.000,- und € 50.000,- pro Jahr an

- Zielerfüllungsfaktor 3
  Es fallen kein oder nur geringe Lizenz- und Supportkosten in Höhe von bis zu € 10.000,- pro Jahr an.

4.5.5 Bewertung Soll-Kriterium 5: Verwaltung der Agenten

- Zielerfüllungsfaktor 0
  Die Alternative bietet keinerlei Verwaltungsunterstützung. Sowohl Agenten-Software Installation/Update, als auch die (lokale) Agenten-Konfiguration sowie die notwendigen Check-Plugins müssen über andere Lösungen erfolgen.

- Zielerfüllungsfaktor 1
  Es gibt zumindest teilweise Verwaltungsunterstützung. Jedoch bieten diese keinen oder nur einen geringen Mehrwert an, da dadurch keine signifikanten Aufwandseinsparungen erzielt werden können.

- Zielerfüllungsfaktor 2
  Die Alternative bietet teilweise Verwaltungsunterstützung, welche einen Mehrwert für die ASFINAG generiert. Durch die Verwaltungsunterstützung können laufend Aufwände reduziert werden.

- Zielerfüllungsfaktor 3
  Die Alternative bietet eine volle Verwaltungsunterstützung in allen den Windows Monitoring Agenten betreffenden Bereichen an. Es sind keinerlei Verwaltungstätigkeiten mit Dritt-Tools notwendig.

4.5.6 Bewertung Soll-Kriterium 6: Interkompatibilität

- Zielerfüllungsfaktor 0
  Der Agent ist lediglich mit dem Monitoring-Agenten Naemon kompatibel.

- Zielerfüllungsfaktor 1
  Die Alternative ist mit (zumindest einigen) weiteren zustandsbasierten Monitoring Backend Systemen kompatibel. Beispielsweise über das NRPE-Protokoll. Oder über die Zurverfügungstellung einer einfachen Web-API, die ohne viel Aufwand von anderen zustandsbasierten Monitoring Backend Systemen verwendet werden kann.

- Zielerfüllungsfaktor 2
  Der Windows Monitoring Agent ist nicht nur mit weiteren zustandsbasierten Monitoring Backend Systemen kompatibel, sondern auch mit (zumindest einigen) metrikbasierten Monitoringsystemen.

- Zielerfüllungsfaktor 3
  Der Agent unterstützt zusätzlich (zumindest teilweise) einen Kompatibilitäts-standard wie beispielsweise OpenTelemetry.

4.5.7 Gesamtübersicht der Kriterien-Bewertungen

Die Bewertungen der Soll-Kriterien sowie die jeweilige Zuordnung zu Zielerfüllungsfaktoren ist zusammengefasst in Tabelle 2 ersichtlich.

		Zielerfüllungsfaktor			
Soll Kriterium	Kurzbezeichnung	0	1	2	3
1	Kompatibilität mit Naemon	Nicht kompatibel. Umstieg auf anderes Monitoringbackend notwendig.	Teilweise kompatibel. Viele Apassungen am Monitoring Backend naemon notwendig.	Weitgehend kompatibel. Einige Apassungen am Monitoring Backend naemon notwendig.	Voll kompatibel. (Fast) keine Anpassungen an Backend Monitoring naemon notwendig.
2	Weiterverwendung individueller Check-Plugins	Vollständige Neuentwicklung notwendig.	Aufwendige Anpassungen notwendig.	Geringe Anpassungen notwendig.	Keinerlei Anpassungen notwendig.
3	Supportunterstützung	Keinerlei Supportunterstützung für den Agenten verfügbar.	Nur rudimentäre kommerzielle Supportuntestützung wird angeboten	Kommerzielle Supportunterstützung wird angeboten.	Umfangreiche kommerzielle Supportunterstützung des Agenten samt SLA.
4	Kosten für Lizenzierung und Support	Sehr hohe Lizenz- und Supportkosten (> € 100.000 pro Jahr)	Hohe Lizenz- und Supportkosten (zw. € 50.000 und € 100.000 pro Jahr)	Mittelhohe Lizenz- und Supportkosten (zw. € 10.000 und € 50.000 pro Jahr)	Keine oder nur sehr geringe Lizenzierungs- und Supportkosten. (< € 10.000 pro Jahr)
5	Verwaltung der Agenten	Keinerlei Verwaltungsunterstützung.	Teilweise Verwaltungsunterstützung ohne bzw. mit nur geringem Mehrwert	Teilweise Verwaltungsunterstützung mit Mehrwerten.	Volle Verwaltungsunterstützung (mehrstufige Konfiguration, Verteilung von individuellen Check Plugins).
6	Interkompatibilität	Nur mit Monitoing Backend naemon kompatibel.	Mit weiteren zustandsbasierten Monitoring Backend Systemen kompatibel.	Mit weiteren zustands- als auch metrikbasierten Monitoringsystemen kompatibel.	Unterstützt einen Kompatibilitätsstandard wie beispielsweiese Open-Telemetry.

Tabelle 2: Bewertung von Soll-Kriterien und Zielerfüllungsfaktoren

4.6 Berechnung des Nutzwertes

Die Berechnung des jeweiligen Nutzwertes einer jeden Lösungsalternative erfolgt in zwei Schritten:

- Berechnung eines Nutzwertes für jedes Soll-Kriterium durch Multiplikation der Zielerfüllungsfaktors mit dem Bewertungsfaktor
- Summierung zu einem gesamten Nutzwert je Alternative

Alternativen, welche zumindest ein Muss-Kriterium nicht erfüllen, werden ausgeschieden.

Eine beispielhafte Nutzwertberechnung ist aus Tabelle 3 ersichtlich.

			Alternative 1		**Alternative 1**		**Alternative x**	
Muss Kriterium	**Kurzbezeichnung**		Zielerfüllung		Zielerfüllung		Zielerfüllung	
1	Weiterentwicklung des Agenten		erfüllt	-	erfüllt	-	erfüllt	-
2	Unterstützung individueller Checks		nicht erfüllt	-	erfüllt	-	erfüllt	-
Soll Kriterium	**Kurzbezeichnung**	**Bewertungsfaktor**	Zielerfüllung	Nutzwert	Zielerfüllung	Nutzwert	Zielerfüllung	Nutzwert
1	Kompatibilität mit Naemon	0,25	3	0,75	1	0,25	0	0
2	Weiterverwendung individueller Check-Plugins	0,25	2	0,5	1	0,25	1	0,25
3	Supportunterstützung	0,15	2	0,3	2	0,3	2	0,3
4	Kosten für Lizenzierung und Support	0,25	2	0,5	2	0,5	1	0,25
5	Verwaltung der Agenten	0,05	0	0	1	0,05	1	0,05
6	Interkompatibilität	0,05	1	0,05	0	0	0	0
	Gesamt	**1,00**	ausgeschieden	~~**2,1**~~		**1,35**		**0,85**

Tabelle 3: Beispielhafte Nutzwertberechnung

5. Vorstellung der sechs alternativen Windows Monitoring Agent Produkte und Durchführung der Nutzwertanalyse

In diesem Kapitel werden die zu untersuchenden Alternativen vorgestellt. Weiters wird für jede Alternative auf die Kriterien eingegangen sowie auf deren Zielerfüllungsfaktor hin untersucht. Anschließend die Nutzwertanalyse durchgeführt.

5.1 Centreon NSClient++

Beschreibung der Alternative Centreon NSClient++ sowie Untersuchung dieser Alternative bezüglich der Kriterien für die Nutzwertanalyse.

5.1.1 Kurzbeschreibung Centreon NSClient++

Im Jahr 2005 wurde das Unternehmen Merethis gegründet. Dieses hatte die Aufgabe die Koordinierung der open-source Community rund um die auf Nagios basierenden open-source Monitoringlösung Oreon3 durchzuführen. Wegen eines Namenskonflikts mit der Monitoringlösung Oreon von Hersteller SolarWinds, wurde 2005 eine Umbenennung der Produktlösung zu Centreon durchgeführt. Das Unternehmen Merethis wurde 2015 in Centreon umbenannt.

In den ersten Jahren ist der Fokus auf Erweiterungen der Nagios-Komponenten gelegen, beispielsweise Reporting. So gibt es auch heute noch ein Modul zu SLA-Auswertungen. Nach und nach wurden die Nagios-Komponenten durch Eigenentwicklungen ersetzt. Schließlich wurde 2013 jede Nagios-Bezeichnung aus Centreon entfernt.

Heute bietet Centreon neben einer kostenlosen open-source Monitoring Basislösung auch umfangreiche kommerzielle Monitoringlösungen mit Supportoptionen an.

Einen Schwerpunkt hat Centreon auf sogenannte Check-Plugins gelegt und diese in sognannte Plugin-Packs organisiert. Diese Plugin-Packs bieten Monitoring-Checks zu unterschiedlichen Systemen an. Etwa zum Monitoring von Datenbanksystemen, Hardware wie Storage, Virtualisierungsumgebungen, Backup-Lösungen, DHCP-Server usw. [29]

Dabei werden Konfigurationstemplates am Monitoring-Backendsystem bereitgestellt. Je nach Art des Monitorings, werden die dazugehörigen eigentlich auszuführenden Check-Plugins auf dem Monitoring-Backendsystem deployt und direkt von dort ausgeführt oder diese Check-Plugins werden über einen Monitoring-Agenten dezentral ausgeführt.

Für Windows-Systeme setzt Centreon dabei auf NSClient++ auf. Im Jahr 2020 hat Centreon die open-source Windows Monitoring Agenten Lösung NSClient++ geforkt und veröffentlicht seither eine angepasste eigene Variante von NSClient++ [7].

All diese Check-Plugins folgen den Richtlinien der Monitoring Plugins Development Guidelines [30]

5.1.2 Kriterienuntersuchung zu Centreon NSClient++

Centreon veröffentlicht alle paar Monate eine aktualisierte Version von Centreon NSClient++. In den letzten 12 Monaten wurden zwei Releases veröffentlicht und es gibt keinerlei Anzeichen, dass Centreon eine Einstellung von deren NSClient++ Variante plant.

Allerdings wird lediglich das Windows Installationspackaging aktualisiert. Im Wesentlichen kommen zusätzliche Check-Plugins hinzu. Die eigentliche Agentensoftware wurde von Centreon bisher nicht angepasst oder weiterentwickelt. Die eigentliche Agentensoftware hat noch immer denselben Stand wie das orginale NSClient++

Im Rahmen der Markterkundung der Fa. Cancom hat ein Austausch mit einem Centreon-Vertriebsmitarbeiter für die DACH-Region stattgefunden. Dabei hat Centreon erklärt, dass NSClient++ kein Centreon Produkt ist und nicht von Centreon gewartet und aktualisiert wird. Man biete den Centreon Kunden lediglich die Möglichkeit, diese Agenten - paketiert mit einer erweiterten Plugin-Sammlung – herunterzuladen und zu verwenden.

Als Hintergrund dazu wurde mündlich genannt, dass der C++ Quellcode von NSClient++ sehr unstrukturiert und komplex wäre und eine Pflege und Erweiterung dieser Quellcodebasis für Centreon nicht möglich sei.

Da die Centreon Variante von NSClient++ auf einer unveränderten Codebasis vom orginalen NSClient++ aufsetzt, werden wie bei NSClient++ individuelle Checks unterstützt.

Centreon nutzt dieses Feature und paketiert in seiner Variante viele zusätzliche individuelle Checks mit dazu.

Da die Centreon Variante von NSClient++ auf einer unveränderten Codebasis vom orginalen NSClient++ aufsetzt, werden wie bei NSClient++ dieselben Schnittstellen und Parametrisierungen verwendet. Es müssen somit am Monitoring Backend keinerlei Konfigurationsanpassungen vorgenommen werden.

Da die Centreon Variante von NSClient++ auf einer unveränderten Codebasis vom orginalen NSClient++ aufsetzt, werden wie bei NSClient++ alle Check-Plugins unterstützt, welche den Monitoring Plugin Development Guidlines entsprechen. Somit können alle bisher eingesetzten Check-Plugins ohne Anpassungen weiterverwendet werden.

Die Fa. Centreon hat erklärt, keine Wartung und Pflege der eigentlichen Agentensoftware NSClient++ durchzuführen. Für die Centreon Variante von NSClient++ wird vom Unternehmen Centreon lediglich ein Support in Form von Installations- und Konfigurationsunterstützung auf Regiestundenbasis angeboten.

Für die eigentliche Agentensoftware NSClient++ wird lediglich angeboten über github einen Issue zu melden und zu schauen, ob sich wer aus der open-source Community dem gemeldeten Problem auf freiwilliger Basis annimmt.

Die Centreon Variante von NSClient++ wird nicht unter einer kommerziellen Lizenz angeboten, sondern wird ausschließlich über eine kostenfreie open-source Lizenzierung zur Verfügung gestellt. Weiters werden keine kommerziellen Support-Optionen für Centreon NSClient++ angeboten.

Auf Regiestundenbasis ist Unterstützungsdienstleistung zu Installation und Konfiguration von Centreon NSClient++ erhältlich. Jedoch wird kein SLA für Entstörungsleistungen im Zusammenhang mit Centreon NSClient++ angeboten.

Wie beim Orginal sind keinerlei Deployment- bzw. Updatemöglichkeiten bei der Agentensoftware enthalten. Centreon paketiert zwar ihre Plugin-Sammlung, jedoch bringt dies keine Möglichkeit mit, um eigene Check-Plugins darüber zu verwalten. Weiters gibt es auch keinerlei Möglichkeiten die (lokale) Agenten-Konfiguration (zentral) zu verwalten.

Für die Centreon Variante von NSClient++ gilt dasselbe wie für das Orginal: Der Agent ist mit dem Naemon Monitoring-Backendsystem kompatibel.

Dieser Monitoring-Agent hat das NRPE-Protokoll und somit eine pull-Abfragemöglichkeit implementiert. Weiters bietet dieser Monitoring-Agent die Möglichkeit über eine Web-API angesprochen zu werden. Andere Monitoring (Backen) Lösungen können somit diesen Agenten ansprechen und einsetzen.

Weitere Interkompatibilitätsmöglichkeiten (z.B. Prometheus-basierende Metriken) sind jedoch keine vorhanden.

5.1.3 Bewertungszusammenfassung zu Centreon NSClient++

Muss Kriterium	Kurzbezeichnung	Zielerfüllung
1	Weiterentwicklung des Agenten	nicht erfüllt
2	Unterstützung individueller Checks	erfüllt

Tabelle 4: Bewertung Muss-Kriterien von Centreon NSClient++

Soll Kriterium	Kurzbezeichnung	Zielerfüllungsfaktor
1	Kompatibilität mit Naemon	3
2	Weiterverwendung individueller Check-Plugins	3
3	Supportunterstützung	1
4	Kosten für Lizenzierung und Support	3
5	Verwaltung der Agenten	1
6	Interkompatibilität	1

Tabelle 5: Bewertung Soll-Kriterien von Centreon NSClient++

Der Centreon Fork von NSClient++ hat sich mehr als Repackaging herausgestellt, als ein klassischer Fork mit eigenständiger Weiterentwicklung der Software.

Die Weiterentwicklung der Agentensoftware ist somit wie beim Orginal nicht gegeben. Dies resultiert in eine Nichterfüllung des Muss-Kriteriums der aktiven Weiterentwicklung des Agenten.

5.2 checkmk

Beschreibung der Alternative checkmk sowie Untersuchung dieser Alternative bezüglich der Kriterien für die Nutzwertanalyse.

5.2.1 Kurzbeschreibung checkmk

Im Jahr 2008 wurde von Matthias Kettner die Nagios Erweiterung Check_MK entwickelt [8].

Die Produktbezeichnung hat sich über die Jahre leicht verändert. Von Check_MK über CheckMK zum heutigen checkmk. Das dahinterliegende Unternehmen war zuerst die Matthias Kettner GmbH, zwischenzeitlich die tribe29 GmbH und seit Mai 2023 firmiert das Unternehmen hinter checkmk unter dem Firmenwortlaut Checkmk GmbH.

Erst waren es einzelne Erweiterungen für Nagios. Später Zusammenstellungen von Tools im Nagios Monitoringumfeld zu einem für Nutzer einfach aufzusetzenden Monitoring-Gesamtsystem.

Die beiden Münchner Unternehmen im open-source Monitoring-Umfeld, Matthias Kettner und Consol, sowie weitere Entwickler haben sich 2010 zusammengetan und die „Open Monitoring Distribution“ (kurz: OMD) ins Leben gerufen. Diese umfasst ein vorkompiliertes und vorkonfiguriertes Nagios-Monitoringsystem samt einem Set an zusätzlichen optionalen Tools. Im Jahr 2017 hat checkmk die gemeinschaftliche Zusammenarbeit mit OMD beendet und seine eigene OMD-Version mit checkmk entwickelt ohne vollständige Kompatibilität zu allen bisherigen anderen OMD Monitoring-Tools. Nach und nach hat sich checkmk zu einem eigenständigen Monitoringgesamtsystem entwickelt. [31]

Seit Beginn von checkmk lag ein Schwerpunkt auf einer guten und einfachen Integration für das Monitoring von klassischen Windows-Systemen. Bereits seit 2008 gibt es einen eigenen Windows Monitoring-Agenten von checkmk.

Die checkmk Lösung ist als Gesamtmonitoringlösung konzipiert. Einzelne Komponenten wie der Windows Monitoring Agent sind nicht einzeln paketiert und downloadbar. Es muss die gesamte (Backend) Lösung heruntergeladen und installiert werden. Von einem eingerichteten checkmk Monitoringbackendsystem kann dann eine Windows Monitoring Agenten msi-Installationsdatei heruntergeladen werden.

5.2.2 Kriterienuntersuchung zu checkmk

Der Windows Monitoringagent von checkmk ist nicht als Einzelkomponente paketiert und verfügbar. Das checkmk Gesamtmonitoringpaket – in dem auch der Windows Monitoring Agent enthalten ist – erhält alle paar Wochen ein Update. Siehe dazu das checkmk Download-Archiv [32]

Die Windows Monitoring Agenten Software wird aktuell gepflegt. Software-Änderungen bzw. Erweiterungen, welche sich auf Nutzer auswirken, werden bei checkmk „Werks“ genannt. Siehe die checkmk Werks-Übersicht [33]

Checkmk bietet die Möglichkeit, für den Windows Monitoring Agenten eigene individuelle Check-Plugins zu entwickeln. Dafür stellt checkmk eine eigene „Check API“ zur Verfügung, auf der aufbauend eigene Check-Plugins entwickelt werden können.

Siehe die checkmk Dokumentation zur Entwicklung von Check-Plugins [34].

Der bisher zum Einsatz gekommene Lösungsansatz mit Naemon als Monitoring-Backend und NSClient++ als Windows Monitoring Agent arbeitet nach den Konzepten des „Aktiven Monitorings“ sowie „Verteilte Ermittlung von Zuständen“ (siehe Kapitel 2.2 und 2.3).

Das bedeutet, dass das Monitoring-Backend naemon einen auf dem Backend antriggert, der Agent vom Backend aktiv angefragt wird und der Agent daraufhin den eigentlichen Check durchführt und auswertet und das ausgewertete Check-Ergebnis an das Monitoring-Backend zurück liefert.

Bei checkmk hingegen basiert das Monitoring von Windows-Systemen auf Konzepten des „Passiven Monitoring“ sowie „Zentrale Ermittlung von Zuständen“ (siehe Kapitel 2.2 und 2.3).

Der checkmk Windows Monitoring Agent triggert autonom die Ausführung von einzelnen Checks auf dem Windows-System. Die ermittelten Rohdaten der Checks werden regelmäßig gesammelt an das checkmk Backendsystem gesendet. Das checkmk Backendsystem wertet dann die Check-Rohdaten zu Check-Ergebnissen aus.

Durch die Nutzung dieser unterschiedlichen Konzepte liegt es auf der Hand, dass diese nicht so ohne weiteres miteinander kompatibel sind. In der Historie war checkmk einst eine Erweiterung im Nagios-Umfeld und wurde auch in der OMD-Distribution gemeinsam mit Naemon paketiert. Eine Kompatibilität gab es daher in der Vergangenheit bereits.

Im Wesentlichen wird die Kompatibilität über ein Modul auf dem Monitoringbackend erreicht. Dieses Modul ist zwischen dem Nagios/Naemon-Backend sowie dem (checkmk) Windows Monitoring Agenten zwischen geschalten.

Mit der Auflösung der Weiterentwicklung der gemeinsamen OMD Distribution im Jahr 2017 gibt es keine Kompatibilität mehr zwischen dem checkmk Monitoringbackend-Zwischenmodul und Naemon.

Weiters wurde inzwischen auch eine Registrierungsfunktion des checkmk-Agenten mit dem checkmk-Backend eingeführt. Nachdem sich der Agent beim Backend gemeldet hat und dieser am Backend bestätigt wurde, wird über Zertifikatsaustausch die Kommunikation abgesichert. Diese Registrierungserweiterung ist checkmk-proprietär und nicht mit einen Naemon-Backend nutzbar.

Die comNET GmbH hat 2018 versucht die Kompatibilität zwischen dem checkmk Windows Monitoring Agenten und Naemon mit einem von checkmk unabhängigen open-source Entwicklungsprojekt herzustellen [35]. Sowie auch in die Monitoring Web-UI Thruk zu integrieren [36].

Diese Initiative wurde auf der OMSC (Open Source Monitoring Conference) 2018 von Fabian Binder unter dem Vortrage mit dem Titel „Integrating Check_MK agent into Thruk – Windows monitoring made easy“ vorgestellt [37]

Jedoch hat dieses Projekt keinen großen Anklang und weitere Mitstreiter gefunden. Inzwischen ist die comNET GmbH ein Partner der höchsten Partnerschaftsstufe von der Checkmk GmbH und setzt vollständig auf die checkmk Monitoringgesamtlösung. Es gibt daher keine Bestrebungen der comNET mehr, eine Kompatibilität zwischen dem checkmk Windows Monitoring Agenten und der verbliebenen OMD-Distribution und Naemon/Thruk wieder herzustellen.

Ergänzend sei festgehalten, dass der checkmk Windows Monitoring Agent inzwischen auch mit einer Pull-Variante ausgestattet wurde. Jedoch wurde zwischenzeitlich auch ein Agenten-Registrierungsmechanismus eingeführt, der nur über die Push-Variante funktioniert und mit dem checkmk Backendsystem verkoppelt ist. Die Pull-Variante holt sich die Check-Rohdaten und leitet diese dem checkmk Backend zur Auswertung weiter.

Eine Anfrage von Cancom beim checkmk Hersteller hat ergeben, dass es keinerlei Bestrebungen gibt, den checkmk Windows Monitoring Agenten für anderen Monitoring Backendsystemen interoperatibel zu machen.

Zusammenfassend kann daher gesagt werden, dass es keinerlei Kompatibilität zwischen dem checkmk Windows Monitoring Agenten und dem Naemon Monitoring Backendsystem (mehr) gibt.

Die Check-Plugins des checkmk Windows Monitoring Agenten müssen eine checkmk „Check API“ verwenden. Dabei wird von checkmk Powershell unterstützt.

Check-Plugins, die den Monitoring Plugin Development Guidlines entsprechen, sind nicht mit dem checkmk Windows Monitoring Agenten kompatibel. Dies liegt darin begründet, dass die eigentliche Checkauswertung bei checkmk nicht beim Check-Plugin liegt, sondern die Aufgabe des Monitoring Backendsystems ist.

Alle vorhandenen individuell entwickelten Powershell Check-Plugins müssten daher aufwendig angepasst werden. Es können lediglich einzelne logische Codeteile übernommen werden.

Checkmk bietet für seine Monitoringgesamtlösung kommerziellen Support an. Für den Windows Monitoring Agenten allein wird keine Supportoption angeboten.

Die angebotenen Supportoptionen sehen jedoch keinen SLA für die Behebung von Bugs und kritischen Schwachstellen des Windows Monitoring Agenten vor.

Um den checkmk Windows Monitoring Agenten nutzen zu können, müsste das gesamte IT-Monitoring von Windows-Systemen der ASFINAG auf checkmk umgestellt werden. Das Monitoring von Windows-Systemen macht etwa 15% des gesamten IT-Monitoringumfangs aus.

Dabei müsste die ASFINAG dann zwei Monitoringlösungen im klassischen Systemmonitoring betreiben und pflegen. Sowie die checkmk Monitoringlösung in das vorhandene Umbrella-Monitoring über eine Schnittstelle anbinden.

Eine entsprechende Anfrage bei checkmk hat ergeben, dass sich checkmk als Umbrella Monitoringsystem platziert und keine Unterstützung für die Integration von checkmk in ein vorhandenes Umbrella-Monitoringsystem anbieten möchte.

Alternativ müsste die ASFINAG das gesamte vorhandene IT (System) Monitoring auf checkmk umstellen. Dies wäre mit erheblichen Lizenzkosten sowie Transitionsaufwand verbunden.

Eine grobe Überschlagsrechnung hat ergeben, dass mit zumindest einem hohen fünfstelligen Eurobetrag pro Jahr für die Lizenzierung samt Support zu rechnen ist. Sowie mit einem weiteren jährlichen Bedarf an Unterstützungsdienstleistungen auf Regiestundenbasis in zumindest niedrigem fünfstelligen Eurobereich. Mit zusätzlich anteilig über mehrere Jahre verteilten Transitionskosten ist von einem Kostenaufwand von mehr als € 100.000,- jährlich auszugehen,.

Checkmk bietet über deren Konfigurations-Webportal des Backendsystems eine sogenannte Agenten-„Backery“ an. Über diese können angepasste msi-Installationspakete erstellt werden die dann von Softwareverteilungslösungen wie SCCM verwendet werden können.

Die msi-Installationspakete vom check Windows Monitoring Agenten sind so vorkonfiguriert, dass sich diese am checkmk-Backend melden und sich von dort ihre Agenten-Konfiguration holen.

Die eigentliche Agenten-Konfiguration erfolgt zentral am checkmk-Backendsystem.

Jedoch können eigene, individuelle Check-Plugins nicht über diesen Mechanismus (nach)verteilt werden. Ist der Einsatz eines individuellen Check-Plugins notwendig, so muss dies über eine eigene Dritt-Softwareverteilungslösung abgewickelt werden.

Checkmk ist die Lösung mit der am weitesten entwickelten Möglichkeiten zur Verwaltung der Windows Monitoring Agenten. Weiters hat checkmk angekündigt in diesem Bereich weiterzuentwickeln und zukünftig zusätzliche Mehrwerte anbieten zu wollen.

Der Checkmk Windows Monitoring Agent ist weder zu Naemon noch mit anderen Monitoringsystemen kompatibel.

Dies ist von checkmk auch nicht gewollt und es gibt (zumindest derzeit) keinerlei Bestrebungen des Herstellers dies zukünftig zu ändern.

5.2.3 Bewertungszusammenfassung zu checkmk

Muss Kriterium	Kurzbezeichnung	Zielerfüllung
1	Weiterentwicklung des Agenten	erfüllt
2	Unterstützung individueller Checks	erfüllt

Tabelle 6: Bewertung Muss-Kriterien von checkmk

Soll Kriterium	Kurzbezeichnung	Zielerfüllungs-faktor
1	Kompatibilität mit Naemon	0
2	Weiterverwendung individueller Check-Plugins	1
3	Supportunterstützung	2
4	Kosten für Lizenzierung und Support	0
5	Verwaltung der Agenten	2
6	Interkompatibilität	0

Tabelle 7: Bewertung Soll-Kriterien von checkmk

Checkmk hat einen sehr guten Gesamteindruck hinterlassen, was seinen Agenten betrifft. Die Integration mit dem checkmk Backend und die zum Teil automatische Erstellung der Monitoring-Backendkonfiguration als auch die (zentralen) Verwaltungsmöglichkeiten des Monitoring-Agenten und die Agenten „Backery“ haben bei einer Demonstration des Herstellers einen sehr positiven Eindruck hinterlassen.

checkmk ist die Lösung mit dem am weitreichendsten Möglichkeiten der zentralen Agentenverwaltung. Dennoch sind nicht alle dafür benötigten Funktionalitäten abgedeckt und man kommt um eine Softwareverteilungsdrittlösung (zur Verteilung individueller Check-Plugins) nicht herum.

Wegen der Inkompatibilität zu Naemon müsste das gesamte Monitoring der Windowssysteme auf eine checkmk Gesamtlösung samt checkmk-Backend erfolgen. Was mit einem erheblichen Transitionsaufwand verbunden ist.

Da der Hersteller eine Integration seiner Monitoringgesamtlösung über eine Schnittstelle in ein bestehendes Monitoring-Umbrellasystem nicht unterstützen möchte, müsste das gesamte klassische IT-Monitoring mit der checkmk Lösung abgelöst werden. Dies würde die Kosten aufgrund des checkmk Lizenzierungsmodells als auch mit dem damit verbundenen sehr hohen Transitionsaufwand in die Höhe schießen.

Weiters können die vorhandenen individuellen Check-Plugins die derzeit mit NSClient++ zum Einsatz kommen nicht übernommen werden. Alle Check-Plugins müssen auf eine checkmk API angepasst werden.

5.3 NRDP

Beschreibung der Alternative NRDP sowie Untersuchung dieser Alternative bezüglich der Kriterien für die Nutzwertanalyse.

5.3.1 Kurzbeschreibung NRDP

Nagios wurde 1999 von Ethan Galstad entwickelt und ist in der IT-Welt sehr bekannt. Daraus entstand auch das Unternehmen Nagios Enterprises LLC

NRDP steht für Nagios Remote Data Processor [9] und ist ein Modul, mit welchem beispielsweise Check-Ergebnisse von Agenten verarbeitetet und an das Nagios-Backendsystem weitergegeben werden können. So kann ein „Passives Monitoring“ wie in Kapitel 2.2 beschrieben realisiert werden.

Jedoch ist NRDP kein Windows Monitoring Produkt. NRDP kommt jedoch häufig in Kombination mit Nagios NCPA zum Einsatz. NCPA ist die Abkürzung von „Nagios Cross-Plattform Agent“. Dies ist ein Monitoring Agentenprodukt für die Plattformen Windows, Linux und Mac. NCPA wird seit 2014 entwickelt. [38]

Im weiteren Verlauf wird daher das Nagios NCPA Montioring Agentenprodukt untersucht.

5.3.2 Kriterienuntersuchung zu NRDP (NCPA)

Über die Download Website von Nagios Enterprises als auch über das github Repository von ncpa ist ersichtlich, dass NCPA aktiv weiterentwickelt wird und es in letzter Zeit mehrere Releases gegeben hat [39] [40]

Eigene Check-Plugins werden von NCPA unterstützt. Die Handhabung ist über einen Support-Artikel von Nagios Enterprises beschrieben [41]

Aus technischer Sicht ist Nagios NCPA mit Naemon kompatibel einsetzbar. Statt über das NRPE-Protokoll und check_nrpe wird auf dem naemon Monitoring Backend das mit Nagios NCPA bereitgestellte check_ncpa.py Plugin aufgerufen. [42] [43]

Der Vollständigkeit halber sei angemerkt, dass die Push-Variante mit NRDP als Backend-Modul mit Naemon technisch nicht kompatibel ist. NRPE funktioniert nur mit dem Nagios-Core, jedoch nicht mit dem Naemon-Core. Nachdem jedoch eine Ablöse von NSClient++ in der Pull-Variante gefragt ist, ist dies für die Bewertung unerheblich.

Aus rechtlicher Sicht gibt es jedoch ein Problem:

Während die meisten Software-Komponenten im Umfeld der Nagios Enterprises der open-source Lizenz GPL unterstellt sind, ist Nagios NCPA unter die „Nagios Community Software License Version 1.3“ gestellt. [44]

Dies ist eine sehr restriktive open-source Lizenzierung. Der Quellcode darf zwar eingesehen werden, jedoch darf dieser nicht verändert werden. Weiters darf die Software unter dieser Lizenz weder für sich allein („standalonoe“) noch in Verbindung mit anderer Software als vom Hersteller Nagios Enterprises verwendet werden. Eine Nutzung von NCPA zusammen mit einem Naemon Monitoring Backend ist daher aufgrund der gewählten open-source Lizenzierung von NCPA nicht erlaubt.

Eine Anfrage an den Hersteller Nagios Enterprises über einen Europa-Kontakt in den Niederlanden (Conclusion Xforce), die Nagios NCPA Agentensoftware über eine kommerzielle Lizenzierung mit Naemon einsetzen zu dürfen, ist nicht positiv beantwortet worden.

Die Check-Plugins für Nagios NCPA müssen den „Nagios Plugins Development Guidelines“ entsprechen [45].

Die vorhandenen Check-Plugins entsprechen den „Monitoring Plugins Development Guidelines“ [30].

Zweitere sind aus den ersteren entstanden und sind miteinander kompatibel. Somit können alle bisher eingesetzten Check-Plugins ohne Anpassungen weiterverwendet werden.

Nagios Enterprises bietet für seine Gesamtmonitoringlösung „Nagios XI“ an. Ein kommerzieller Support für den Windows Monitoring Agenten Nagios NCPA wird über die Unternehmenswebsite nicht angeboten.

Auf eine diesbezügliche Anfrage an den Europa-Partner Conclusion Xforce kam keine positive Rückmeldung zurück.

Allgemeine Installations- und Konfigurationsunterstützung des NCPA-Agenten (auf Regiestundenbasis) ist jedenfalls über Partner von Nagios Enterprises erhältlich.

Weiters kann beim Hersteller die Gesamtmonitoringlösung „Nagios XI“ lizenziert werden und Probleme bzw. Bugs an der Agentensoftware NCPA gemeldet werden. Eine SLA ist dafür jedoch nicht erhältlich.

Die Situation stellt sich ähnlich wie bei checkmk dar.

Bei Verwendung von Nagios NCPA als Windows Monitoring Agentenlösung müsste auch das IT-Monitoring Backend der ASFINAG für das Monitoring von Windows-Systemen auf die Gesamtlösung „Nagios XI“ umgestellt werden.

Dabei würden zwei Monitoringlösungen parallel betrieben werden und die „Nagios XI“ (mit Naigos NCPA Windows Monitoring Agenten“ über eine Schnittstelle an das vorhandene Umbrella-Monitoring angebunden werden.

Wie bei checkmk ist auch der Hersteller Nagios Enterprises über solch einen Mischbetriebsansatz nicht besonders glücklich. Die Kommunikation mit Nagios Enterprises über deren Europapartner Conclusion Xforce gestaltete sich sehr schwierig und ist schließlich im Sande verlaufen.

Alternativ müsste die ASFINAG das gesamte vorhandene IT (System) Monitoring auf die Produktlösungen von Nagios Enterprises umstellen. Dies wäre mit erheblichen Lizenzkosten sowie Transitionsaufwand verbunden.

Eine grobe Überschlagsrechnung hat ergeben, dass mit einem mittleren fünfstelligen Eurobetrag pro Jahr für die Lizenzierung samt Support zu rechnen ist. Sowie mit einem weiteren jährlichen Bedarf an Unterstützungsdienstleistungen auf Regiestundenbasis in zumindest niedrigem fünfstelligen Eurobereich. Mit zusätzlich anteilig über mehrere Jahre verteilten Transitionskosten ist mit einem Kostenaufwand von einem hohen fünfstelligen Eurobetrag jährlich auszugehen,

Für den Windows Monitoring Agenten Nagios NCPA bietet Nagios Enterprises keinerlei Unterstützung bezüglich zentraler Verwaltung an.

Sowohl die Agenten-Software Installation/Update, als auch die (lokale) Agenten-Konfiguration sowie die Verteilung der notwendigen individuellen Check-Plugins müssen über andere Lösungen erfolgen.

Bereits wegen der Lizenzierung ist eine Interkompatibilität zu anderen Monitoringlösungen rechtlich ausgeschlossen. Auch wenn dies zumindest teilweise technisch möglich wäre.

5.3.3 Bewertungszusammenfassung zu NRDP (NCPA)

Muss Kriterium	Kurzbezeichnung	Zielerfüllung
1	Weiterentwicklung des Agenten	erfüllt
2	Unterstützung individueller Checks	erfüllt

Tabelle 8: Bewertung Muss-Kriterien von NRDP (NCPA)

Soll Kriterium	Kurzbezeichnung	Zielerfüllungs-faktor
1	Kompatibilität mit Naemon	0
2	Weiterverwendung individueller Check-Plugins	3
3	Supportunterstützung	2
4	Kosten für Lizenzierung und Support	1
5	Verwaltung der Agenten	0
6	Interkompatibilität	0

Tabelle 9: Bewertung Soll-Kriterien von NRDP (NCPA)

Der NCPA Monitoring Agent ist technisch kompatibel zu Naemon und den bisher verwendeten individuellen Check-Plugins. Die vorhandene Monitoring Backend Konfiguration müsste allerdings angepasst werden.

Durch die Unternehmenspolitik des Herstellers ist die Verwendung lizenzmäßig jedoch untersagt und damit nicht in Verbindung mit der vorhandenen Monitoringumgebung einsetzbar.

Um den NCPA Agenten einsetzen zu können, müsste das gesamte IT-Monitoring der ASFINAG mit den Monitoringlösungen der Herstellers Nagios Enterprises abgelöst werden. Dies würde die Kosten aufgrund Lizenzierungskosten als auch mit dem damit verbundenen sehr hohen Transitionsaufwand in die Höhe treiben.

Jedoch können die vorhandenen individuellen Check-Plugins die derzeit mit NSClient++ zum Einsatz kommen einfach übernommen werden. Alle vorhandenen Check-Plugins sind auch mit dem NCPA Kompatibel.

5.4 openITCOCKPIT

Beschreibung der Alternative openITCOCKPIT sowie Untersuchung dieser Alternative bezüglich der Kriterien für die Nutzwertanalyse.

5.4.1 Kurzbeschreibung openITCOCKPIT

Die it-novum GmbH ist der Hersteller hinter der open-source Monitoringlösung „openITCOCKPIT“. Der Grundgedanke hinter openITCOCKPIT ist es eine

Gesamtmonitoringlösung zur Verfügung zu stellen. Historisch setzt openITCOCKPIT auf einen Nagios-Kern bzw. mittlerweile auf den davon geforkten Naemon-Kern.

Die it-novum hat im Jahr 2020 mit der Entwicklung eines eigenen Monitoring Agenten begonnen. Die Versionen 1.x und 2.x wurden in der Programmiersprache Python entwickelt. Ab dem Jahr 2021 wurde mit der Version 3.x auf die Programmiersprache Go gewechselt und von vollständig neuentwickelt [46].

5.4.2 Kriterienuntersuchung zu openITCOCKPIT

Die Entwicklung des open-source Monitoring Agenten von openITCOCKPIT ist über github öffentlich einsehbar.

Die Releases sind einsehbar unter [47].

Obwohl die Häufigkeit der Release-Veröffentlichungen im Jahr 2023 zurück gegangen ist, ist erkennbar, dass nach wie vor eine aktive Weiterentwicklung des Monitoring Agenten stattfindet.

Der openITCOCKPIT Monitoring Agent kann individuelle Checks ausführen. Dies ist auf einer eigenen Site auf dem github-Projekt beschrieben [48].

Das Backend der openITCOCKPIT Gesamtmonitoringlösung baut auf einem Naemon-Kern auf. Und der openITCOCKPIT Agent unterstützt sowohl pull- als auch push-Modus (siehe Kapitel 2.2). Wobei openITCOCKPIT den push-Modus bevorzugt.

Push-Modus:

Beim push-Modus übermittelt der Agent die Check-Ergebnisse an das Backend. Die Entgegennahme und Verarbeitung auf Backend-Seite ist im openITCOCKPIT integriert. Obwohl die verarbeiteten Daten schließlich in einem Naemon-Kern landen, ist keine technische Kompatibilität mit anderen Naemon Monitoringbackends gegeben. Da dieses Backend-Modul nicht eigenständig existiert, kann dieses nicht mit anderen Naemon-basierten Monitoringbackends verwendet werden.

Beim push-Modus registriert sich der Agent beim Backend. Wenn dies beim Backend bestätigt wird, dann erfolgt ein Zertifikatsaustausch und der Agent kann dann nur noch mit diesem einen (openITCOCKPIT) Backend verschlüsselt kommunizieren. Dieser Mechanismus erhöht die Betriebssicherheit. Ist jedoch nicht mit anderen Backend-Lösungen wie Naemon kompatibel.

Man müsste daher für das Monitoring der Windows-Systeme ein openITCOCKPIT Backend einsetzen. Über eine Web-API Schnittstelle des openITCOCKPIT Backends können die Monitoring-Ergebnisse der überwachten Windows-Systeme an das vorhandene Naemon basierte Umbrella Monitoringsystem angebunden werden.

Die vorhanden Monitoring Backend Konfiguration müsste daher an die openITOCKPIT Backend Web-API angepasst werden. Weiters muss auf dem openITCOCKPIT Backendsystem die Backend Konfiguration aller Windows-Systeme neu eingerichtet und dort laufend gepflegt werden.

Pull-Modus:

Die Verwendung der pull-Variante des Agenten ist technisch grundsätzlich auch möglich. Der openITCOCKPIT Agent unterstützt jedoch nicht das NRPE-Protokoll, sondern nur seine eigene proprietäre (pull) API Implementierung. Es müsste jedoch ein eigenes Plugin für das vorhandene Naemon-Backend entwickelt werden welches die proprietäre (pull) Web-API des openITCOCKPIT Agenten ansteuert. Diese Eigenentwicklung müsste auch laufend selbst gewartet und gepflegt werden. Etwa wenn sich die Web-API auf Agentenseite ändert.

Der Hersteller it-novum hat Unterstützung bei der Eigenentwicklung zwischen Naemon-Backend und openITCOCKPIT Agent zugesagt. Jedoch wird it-novum diese nicht offiziell supporten und bei der Weiterentwicklung des Agenten bzw. deren Web-API keine Rücksicht auf Kompatibilität diesbezüglich nehmen.

Die vorhandene Backend-Konfiguration für das Monitoring von Windows-Systemen kann in diesem Fall nicht übernommen werden, sondern muss angepasst werden.

Weiters muss einschränkend erwähnt werden, dass zwar Check-Ergebnisse über eine pull Web-API abgerufen werden können, jedoch die die eigentliche Check-Durchführung davon unabhängig vom Agenten durchgeführt wird. Über die Agenten Konfiguration wird festgelegt, wie oft welcher Check mit welcher Parametrisierung ausgeführt wird. Über die pull Web-API können keine Check-Ausführungen angetriggert werden und auch keine Parametrisierung für die Checkausführung vorgenommen werden.

Der openITCOCKPIT Monitoring Agent ist kompatibel mit den Monitoring Plugins Development Guidelines. Alle vorhandenen Check-Plugins werden daher ohne Anpassungen unterstützt.

Die it-novum bietet keine kommerzielle Lizenzierung für deren Agentenlösung an. Auch ein SLA für Supportdienstleistungen für die Agentenlösung wird nicht angeboten.

Jedoch bietet die it-novum allgemeine Consultingdienstleistungen auf Regiestundenbasis an. Primär für Installations- und Konfiguration ihrer Agentenlösung.

Die it-novum hat sich jedoch grundsätzlich offen dafür gezeigt, auf best-effort Basis Supportleistungen für die Agentensoftware zu leisten.

Es gibt hier zwei mögliche Varianten für die ASFINAG:

Variante1:

Verwendung des openITCOCKPIT Backends für das Monitoring von Windows-Systemen samt Nutzung des openITCOCKPIT Agenten in der von it-novum präferierten push-Variante. Die openITCOCKPIT bietet eine Web-API an mit er man die Monitoringwerte der überwachten Windows-Systeme vom openITCOCKPIT Backendsystem in das bei der ASFINAG vorhanden Umbrella-Monitoringsystem übernehmen kann.

Eine grobe Überschlagsrechnung hat ergeben, dass mit einem mittleren fünfstelligen Eurobetrag pro Jahr für die Lizenzierung samt Support zu rechnen ist. Sowie mit einem weiteren jährlichen Bedarf an Unterstützungsdienstleistungen auf Regiestundenbasis in zumindest niedrigem fünfstelligen Eurobereich. Mit zusätzlich anteilig über mehrere Jahre verteilten Transitionskosten ist mit einem Kostenaufwand von einem hohen fünfstelligen Eurobetrag jährlich auszugehen.

Das würde den Zielerfüllungsfaktor 1 ergeben.

Variante 2:

Nutzung des openITCOCKPIT Monitoring Agenten über die kostenfreie open-source Lizenzierung. Sowie Eigenentwicklung und Pflege eines eigenen Backend-Plugins für pull-Abfragen des Agenten. Dafür wird mit einem niedriger fünfstelligen Eurobetrag jährlich gerechnet.

Zusätzlich beziehen eines jährlichen Regiestundenpools bei der it-novum in Höhe eines niedrigen fünfstelligen Eurobetrages.

Weiters fallen auch hier Transitionskosten an, da die vorhandene Backen-Konfiguration nicht übernommen werden kann sondern angepasst werden muss. Jedoch muss keine zweite Monitoring-Backendlösung aufgebaut und laufend gepflegt werden.

Es summiert sich dennoch zu einem jährlichen mittleren fünfstelligen Eurobetrag. Das würde den Zielerfüllungsfaktor 2 ergeben.

Die Variante 2 ist geringerem Aufwand und Kosten für die ASFINAG sinnvoller.

Für den openITCOCKPIT Monitoring Agenten bietet it-novum keinerlei Unterstützung bezüglich zentraler Verwaltung an.

Sowohl die Agenten-Software Installation/Update, als auch die (lokale) Agenten-Konfiguration sowie die Verteilung der notwendigen individuellen Check-Plugins müssen über andere Lösungen erfolgen.

Die it-novum zeigt sich grundsätzlich offen für Integration von weiteren Monitoringlösungen. Das openITCOCKPIT Backend verfügt über eine Integration von checkmk als auch prometheus.

Weiters zeigt sich die it-novum auch offen für Integrationsansätze ihrer Produkte und Lösungen in andere Monitoringlösungen und blockt dies nicht wie andere bisher untersuchte Hersteller kategorisch ab.

Der openITCOCKPIT Agent kann etwa über die vorhandene Web-API von anderen Lösungen in der pull-Variante abgefragt werden. Jedoch ist diese Web-API Schnittstelle nicht normiert/standardisiert und gibt es Seitens it-novum keine Zusage diese Web-API dauerhaft zur Verfügung zu stellen.

Weiters ist die push-Variante nicht mit anderen Monitoring Backendlösungen kompatibel. Dies ist auch nicht geplant.

5.4.3 Bewertungsübersicht zu openITCOCKPIT

Muss Kriterium	Kurzbezeichnung	Zielerfüllung
1	Weiterentwicklung des Agenten	erfüllt
2	Unterstützung individueller Checks	erfüllt

Tabelle 10: Bewertung Muss-Kriterien von openITCOCKPIT

Soll Kriterium	Kurzbezeichnung	Zielerfüllungs-faktor
1	Kompatibilität mit Naemon	1
2	Weiterverwendung individueller Check-Plugins	3
3	Supportunterstützung	2
4	Kosten für Lizenzierung und Support	2
5	Verwaltung der Agenten	0
6	Interkompatibilität	1

Tabelle 11: Bewertung Soll-Kriterien von openITCOCKPIT

Mit dem openITCOCKPIT Windows Agenten ist es möglich, das Monitoring der Windows-Systeme zu implementieren, ohne dabei das gesamte IT-Monitoring ablösen zu müssen.

Weiters können die bestehenden individuellen Check-Plugins die derzeit mit dem NSClient++ eingesetzt werden ohne Anpassungen übernommen werden.

Jedoch muss die Monitoring Backend Konfiguration angepasst werden.

Es sind zwei Varianten möglich:

1. Verwendung der openITCOCKPIT Agenten über push-Modus mit einem openITCOCKPIT Backendsystem und einer Schnittstellenanbindung an das vorhandene Naemon-basierende Umbrella Monitoringsystem.
2. Ansprechen der openITCOCKPIT über deren pull Web-API vom vorhandenen Naemon-Backend aus mittels einer Eigenentwicklung. Diese Variante wird vom Hersteller it-novum zwar durchaus akzeptiert und geduldet, jedoch nicht vom Hersteller supportet.

Die vorhandene Monitoring Backend Konfiguration kann in beiden Varianten nicht übernommen werden, sondern muss angepasst werden.

Die Variante 2 ist in Summer mit weniger Transitionsaufwänden und Gesamtkosten verbunden. Daher wird diese Variante bewertet.

openITCOCKPIT ist die erste der bisher untersuchten Varianten, bei der es nicht notwendig ist, das gesamte IT-Monitoring (Backend) System der ASFINAG vollständig abzulösen. Zudem hat sich der Hersteller offen dafür gezeigt eine Integration in die bereits bestehende IT-Monitoringumgebung der ASFINAG zu unterstützen.

5.5 Icinga

Beschreibung der Alternative Icinga sowie Untersuchung dieser Alternative bezüglich der Kriterien für die Nutzwertanalyse.

5.5.1 Kurzbeschreibung Icinga

Im Jahr 2009 hat sich eine Gruppe von Entwicklern aus dem Nagios-Umfeld dazu entschlossen einen Nagios-Fork mit der Bezeichnung Icinga starten. Später entstand das Unternehmen Icinga GmbH.

Es wurde im Jahr 2012 mit der Entwicklung eines neuen Kerns begonnen. Im Juni 2014 wurde die erste stabile Version von Icinga2 veröffentlicht. Diese ist inkompatibel mit dem Nagios-Kern. Dies wurde bewusst in Kauf genommen, um damit bessere Konfigurationsmöglichkeiten bei größeren Umgebungen zu schaffen.

Icinga hat sich intensiv mit agentenbasiertem Monitoring und verteilten Check-Ausführungen (z.B. auf zu überwachenden Windows-System lokal ausgeführte Checks) beschäftigt. Über die Jahre wurden mehrere Lösungsansätze dazu implementiert.

Aktuell setzt Icinga beim Monitoring von Windows-Systemen auf einen auf PowerShell basierenden selbstentwickelten Monitoring Agenten und nennt diesen „Icinga for Windows“ [49].

Die Installation des Powershell-basierenden Agenten kann über alle Arten durchgeführt werden die Powershell unterstützt. Icinga stellt eine Paketierung deren Windows Agentenlösung über den Paketmanager Chocolatey bereit [50].

5.5.2 Kriterienuntersuchung zu Icinga

Die Entwicklung des open-source Monitoring Agenten von Icinga für Windows ist über github öffentlich einsehbar.

Die Releases sind einsehbar unter https://github.com/Icinga/icinga-powershell-framework/releases/
Es ist erkennbar, dass eine aktive Weiterentwicklung des Monitoring Agenten stattfindet.

Der „Icinga for Windows“ Monitoring Agent kann individuelle Checks ausführen. Dies ist im Icinga Developer Guide zu Custom Plugins beschrieben [51].

Es gibt keinerlei Kompatibilität zu einem Naemon Backend.

Der „Icinga for Windows“ Agent ist ausschließlich mit einem Icinga 2 Monitoringbackendsystem kompatibel. Wie bei checkmk und openITCOCKPIT kommt ein Registrierungsmechanismus zu tragen. Dieser ist so implementiert, dass keine Kompatibilität mit anderen Monitoring Backend Lösungen gegeben ist. Anders als bei openITCOCKPIT gibt es für die Verwendung der implementierten pull Web-API Abfrage durch eine Eigenentwicklung keinerlei Unterstützung seitens des Herstellers. Die Agenten Web-API ist nicht für Benutzung von Dritten ausgelegt.

Obwohl der Icinga Windows Monitoring Agent ebenfalls auf Powershell basiert, ist eine Weiterverwendung der bisherigen individuellen Check-Plugins nicht möglich.

Alle vorhandenen individuell entwickelten Powershell Check-Plugins müssen aufwendig angepasst werden. Es können lediglich einzelne logische Codeteile übernommen werden.

Auch Icinga bietet keine kommerzielle Lizenzierung für deren Agentenlösung an. Ebenso wird kein SLA für Supportdienstleistungen für die Agentenlösung angeboten.

Icinga hat sich grundsätzlich offen dafür gezeigt, auf best-effort Basis Supportleistungen für die Agentensoftware zu leisten. Fehler bzw. Schwachstellen der Agentenlösung können bei Icinga gemeldet werden.

Icinga bietet – auch über Partner – allgemeine Consultingdienstleistungen auf Regiestundenbasis an. Primär für Installations- und Konfiguration ihrer Gesamtlösung in der auch die Agentenlösung miteingeschlossen ist. Der „Icinga for Windows“ Monitoringagent ist ausschließlich mit der Icinga Backendmonitoringlösung einsetzbar.

Um den „Icinga for Windows“ Monitoring Agenten nutzen zu können, müsste wie bei checkmk das gesamte IT-Monitoring von Windows-Systemen der ASFINAG auf Icinga umgestellt werden. Das Monitoring von Windows-Systemen macht etwa 15% des gesamten IT-Monitoringumfangs aus.

Dabei müsste die ASFINAG dann zwei Monitoringlösungen im klassischen Systemmonitoring betreiben und pflegen. Sowie die Icinga Monitoringlösung in das vorhandene Umbrella-Monitoring über eine Schnittstelle anbinden.

Eine entsprechende Anfrage bei Icinga hat ergeben, dass sich solch eine Integration in ein vorhandenes Umbrella-Monitoringsystem über eine Web-API des Icinga Monitoringbackends möglich ist.

Eine grobe Überschlagsrechnung hat ergeben, dass mit zumindest einem hohen fünfstelligen Eurobetrag pro Jahr für die Lizenzierung samt Support zu rechnen ist. Sowie mit einem weiteren jährlichen Bedarf an Unterstützungsdienstleistungen auf Regiestundenbasis in zumindest niedrigem fünfstelligen Eurobereich. Mit zusätzlich anteilig über mehrere Jahre verteilten Transitionskosten ist mit einem Kostenaufwand von mehr als € 100.000,- jährlich auszugehen,.

Für den „Icinga for Windows“ Monitoring Agenten bietet Icinga eine Unterstützung der (lokalen) Agentenkonfiguration über eine zentrale Verwaltung an. Weiters gibt es auch die Möglichkeit über allgemeine Powershell Repo-Möglichkeiten die Agentensoftware zu deployen und upzudaten.

Die Verteilung von individuellen Check-Plugins muss anderweitig über Dritt-Lösungen durchgeführt werden.

Der „Icinga for Windows“ Monitoring Agent ist weder zu Naemon noch mit anderen Monitoringsystemen kompatibel. Dies ist von Icinga auch nicht gewollt und es gibt (zumindest derzeit) keinerlei Bestrebungen des Herstellers dies zukünftig zu ändern.

5.5.3 Bewertungsübersicht zu Icinga

Muss Kriterium	Kurzbezeichnung	Zielerfüllung
1	Weiterentwicklung des Agenten	erfüllt
2	Unterstützung individueller Checks	erfüllt

Tabelle 12: Bewertung Muss-Kriterien von Icinga

Soll Kriterium	Kurzbezeichnung	Zielerfüllungs-faktor
1	Kompatibilität mit Naemon	0
2	Weiterverwendung individueller Check-Plugins	1
3	Supportunterstützung	2
4	Kosten für Lizenzierung und Support	0
5	Verwaltung der Agenten	1
6	Interkompatibilität	0

Tabelle 13: Bewertung Soll-Kriterien von Icinga

Icinga zeigt sich mit seinem Windows Monitoring Agenten „Icinga for Windows“ in vielen Punkten ähnlich zu checkmk.

Die technische Kompatibilität ist durch proprietäre Erweiterungen gehemmt. Anbindung von Drittsystemen an die eigene Agentenlösung ist von Seiten des Herstellers nicht gewollt.

Die vorhandenen Check-Plugins können nicht übernommen werden, sondern müssen aufwendig angepasst werden.

Ein Aufbau eine Icinga-Lösung samt Icinga-Backend für das Monitoring der Windows-Systeme sowie Anbindung über eine Schnittstelle an das vorhandene Umbrella-Monitoringsystem ist technisch möglich. Diese Variante wird im Gegensatz zu checkmk vom Hersteller nicht abgelehnt. Sondern wie bei openITCOCKPIT als mögliche Lösungsoption angesehen.

Es gibt Ansätze zu zentrale Verwaltungsmöglichkeiten des Agenten, jedoch sind diese nicht so ausgeprägt wie bei checkmk.

5.6 SNClient+

Beschreibung der Alternative SNClient+ sowie Untersuchung dieser Alternative bezüglich der Kriterien für die Nutzwertanalyse.

5.6.1 Kurzbeschreibung SNClient+

Die Monitoring Agentenlösung SNClient+ wurde erst im Jahr 2023 als open-source Projekt von der Consol Software GmbH in der Programmiersprache Go gestartet.

Das Unternehmen betreut seit der Abspaltung von checkmk im Wesentlichen allein die „Open Monitoring Distribution“ OMD. Zur Abgrenzung zu checkmk wird diese inzwischen „OMD Labs“ genannt [52] [1].

Die Consol hat bis dahin auf die Windows Monitoring Agentenlösung NSClient++ gesetzt. Nachdem die Weiterentwicklung von NSClient++ nicht mehr gegeben war (letzte Release von 27.04.2018) und NSClient++ mit einer nur alten und inzwischen als unsicher geltenden TLS-Unterstützung (TLSy <= 1.2) ausgestattet ist, gab es aus Sicht von Consol akuten Handlungsbedarf.

Da der gesamte Quellcode von NSClient++ über GPL lizenziert ist und quelloffen über github einsehbar ist, war die erste Überlegung der Consol, dieses open-source Projekt

(wie auch schon von Centreon) zu forken und die akuten Herausforderungen im TLS-Bereich anzupassen.

Die Consol hat mit der Beteiligung am Fork von Naemon von Nagios im Jahr 2015 bereits Erfahrung bei so einem Vorhaben. Jedoch hat sich gezeigt, dass die Codebasis von NSClient++ nur sehr schwer bis gar nicht wartbar ist. Weiters ist auch der Build-Prozess von NSClient++ sehr komplex. Die Consol hat dies jedenfalls nicht geschafft.

Anschließend hat die Consol eine Recherche zu alternativen open-source Monitoring Agentenlösung durchgeführt, welche zu Naemon-Backendsystemen kompatibel ist.

Schlussendlich hat sich die Consol dazu entschlossen eine eigene open-source Monitoring Agentenlösung in der Programmiersprache Go zu entwickeln. Diese wurde auf SNClient+ getauft mit dem Ziel eine eins-zu-eins Ablöse der bisher von der Consol präferierten NSClient++ Monitoringagentenlösung zu ermöglichen. Um dies zu erreichen, war der Consol u.a. wichtig, dass die Naemon Backend Konfiguration gleichbleiben kann, die bisherige NSClient+ Agentenkonfiguration kompatibel bleibt und auch die Check-Plugins samt Aufruf bzw. Parametrisierung gleichbleibt.

Die Namensähnlichkeit war daher bewusst gewählt, um damit die Ablösemöglichkeit zu suggerieren. Mit der Drehung der ersten beiden Buchstaben vom Vorbild ergibt sich die Bezeichnung „Secure Naemon Client“.

Der SNClient+ Monitoirng Agent ist ein Dual-Stack Agent, der einerseits die klassischen NSClient+ Fähigkeiten umfasst und andererseits um Prometheus Fähigkeiten erweitert (Prometheus Exporter für Metriken).

Die erste sehr frühe Entwickler-Release wurde am 02.04.2023 veröffentlicht [53].

Der SNClient+ wurde erstmals am 21.11.2023 im Rahmen der Open Source Monitoring Conferece (OMSC 2023) in Nürnberg einer breiteren Öffentlichkeit vorgestellt [54].

Der SNClient+ befand sich währen der Marktrecherche der Cancom noch in einem sehr frühen Entwicklungsstadium. Es fehlten noch wesentliche Funktionalitäten und war damals noch nicht in einer Produktivumgebung einsetzbar.

5.6.2 Kriterienuntersuchung zu SNClient+

Die Entwicklung des open-source Monitoring Agenten SNClient+ ist über github öffentlich einsehbar. Die Releases von SNClient+ sind bei github unter [53] veröffentlicht.

Es ist erkennbar, dass eine aktive Weiterentwicklung des Monitoring Agenten stattfindet.

Eines der Ziele der Consol bei der Entwicklung von SNClient+ war es, dass wie bei NSClient++ individuelle Check-Plugins möglich sind. Alle individuellen Check-Plugins

die bisher mit NSClient++ funktioniert haben, können auch mit SNClient+ (ohne Anpassungen) eingesetzt werden.

Es gibt eine vollständige Kompatibilität zum Naemon Backend. Neben einer allgemeinen technischen Kompatibilität mit Naemon war es auch eines der Ziele von SNClient+, dass die Naemon Backendkonfiguration bei einem Wechsel von NSClient++ zu SNClient+ gleichbleiben kann. Die Namensgebung „Secure Naemon Client“ soll dies verdeutlichen.

Wie alle anderen bisher untersuchten Lösungen wird auch für SNClient+ vom Hersteller Consol kein kommerzieller Support mit SLA-Vereinbarung angeboten.

Ebenso wie auch die anderen Hersteller, bietet die Consol Supportdienstleistungen für ihre Gesamtmonitoringlösung und insbesondere für das Monitoringbackend Naemon an.

Die Consol zeigt sich offen für Supportdienstleistungen rund um ihren Monitoring Agenten auf Regiestundenbasis sowie best-effort Basis.

Für die Nutzung von SNClient+ fallen keine Lizenzgebühren an. Für angebotene Supportdienstleistungen für den Monitoring Agenten auf Regiestundenbasis wird ein niedriger vierstelliger Eurobetrag pro Jahr angenommen.

Es fallen keine weiteren verpflichtenden Aufwände für Backendlizenzierung oder Transitionskosten für Anpassungen bei Backendkonfiguration an.

Optional ist es möglich für das bei der ASFINAG im Einsatz befindliche OMD/Naemon Backendsystem Supportdienstleistungen (auch mit SLA) bei der Consol zu beziehen.

Für das eigentliche Ansinnen den bisherigen Windows Monitoring Agenden NSClient++ abzulösen, fallen jedoch keine weiteren Kosten als der kalkulierte vierstellige Eurobetrag pro Jahr an.

SNClient+ bietet die Möglichkeit einer Autoupdate Funktion über ein zentrales Repo an. Die lokale Agentenkonfiguration muss dafür entsprechend eingerichtet werden.

Zumindest der Erstdeploy muss immer über eine Softwareverteilungsdrittlösung erfolgen. Die Autoupdate Funktion betrifft auch nur die eigentliche Agentensoftware. Die Anpassung an (individuelle) Agentenkonfigurationen sowie die Verteilung von individuellen Check-Plugins muss wiederum über eigene Lösungen durchgeführt werden.

Neben der Kompatibilität zu einem Naemon-Backend kann der Monitoring Agent SNClient+ auch von weiteren Monitoring (Backend) Lösungen wie Nagios, Icinga, openITCOCKPIT, Centreon genutzt werden. Einerseits über das NRPE-Protokoll und andererseits über eine Web-API Schnittstelle von SNClient+.

Mit dem integrierten Prometheus-Stack (Prometheus Exporter) ist SNClient+ auch mit metrikenbasierenden Monitoring (Backend) Lösungen, die Daten in Prometheus Exporter Format unterstützen, kompatibel.

5.6.3 Bewertungsübersicht zu SNClient+

Muss Kriterium	Kurzbezeichnung	Zielerfüllung
1	Weiterentwicklung des Agenten	erfüllt
2	Unterstützung individueller Checks	erfüllt

Tabelle 14: Bewertung Muss-Kriterien von Icinga

Soll Kriterium	Kurzbezeichnung	Zielerfüllungs-faktor
1	Kompatibilität mit Naemon	3
2	Weiterverwendung individueller Check-Plugins	3
3	Supportunterstützung	2
4	Kosten für Lizenzierung und Support	3
5	Verwaltung der Agenten	1
6	Interkompatibilität	2

Tabelle 15: Bewertung Soll-Kriterien von Icinga

Man merkt, dass die Consol das klare Ziel vor Augen hatte mit ihrem neu entwickelten Agenten einen 1-zu-1 Ersatzes den NSClient++ zu ermöglichen. Alle von der ASFINAG benötigten Funktionalitäten von NSClient++ werden von SNClient+ erfüllt.

Die vorhandenen individuellen Check-Plugins können ohne Anpassungen direkt verwendet werden. Volle Kompatibilität mit Naemon. Die vorhandene Backend-Konfiguration kann ohne Anpassungen übernommen werden.

Der Hersteller bietet bis auf einen SLA auf den Agenten vollen Support an. Die Neuentwicklung ist unter der open-source Lizenzierung GPL lizenzkostenfrei einsatzbar.

Es gibt die Möglichkeit über Angabe eines Repos ein Auto-Update der Agentensoftware einzurichten. Dies stellt eine Erweiterung gegenüber NSClient++ dar. Ansonsten gibt es jedoch keine weiteren Möglichkeiten zur zentralen Verwaltung des Agenten samt Konfiguration und individueller Check-Plugins.

Es ist der einzige Agent aller untersuchten Alternativen, welche mit einem Prometheus-Stack in der Agentensoftware auch für weitere (Metriken-basierende) Monitoring Lösungen einsetzbar ist.

5.7 Übersicht aller Kriterienbewertungen

Die aus den vorangegangenen Kapiteln ermittelten Zielerfüllungen sind in Tabelle 16 übersichtlich zusammengefasst:

		Centreon NSClient++	checkmk	NRDP (NCPA)	openITCOCKPIT	Icinga	SNClient+
Muss Kriterium	**Kurzbezeichnung**	**Zielerfüllung**	**Zielerfüllung**	**Zielerfüllung**	**Zielerfüllung**	**Zielerfüllung**	**Zielerfüllung**
1	Weiterentwicklung des Agenten	nicht erfüllt	erfüllt	erfüllt	erfüllt	erfüllt	erfüllt
2	Unterstützung individueller Checks	erfüllt	erfüllt	erfüllt	erfüllt	erfüllt	erfüllt
Soll Kriterium	**Kurzbezeichnung**	**Zielerfüllungs-faktor**	**Zielerfüllungs-faktor**	**Zielerfüllungs-faktor**	**Zielerfüllungs-faktor**	**Zielerfüllungs-faktor**	**Zielerfüllungs-faktor**
1	Kompatibilität mit Naemon	3	0	0	1	0	3
2	Weiterverwendung individueller Check-Plugins	3	1	3	3	1	3
3	Supportunterstützung	1	2	2	2	2	2
4	Kosten für Lizenzierung und Support	3	0	1	2	0	3
5	Verwaltung der Agenten	1	2	0	0	1	1
6	Interkompatibilität	1	0	0	1	0	2

Tabelle 16: Übersicht aller Bewertungen zu Muss/Soll-Kriterien

5.7.1 Auffälligkeiten bei den Kriterienbewertungen

Bei den Soll Kriterien lassen sich durchschnittliche Zielerfüllungsfaktorwerte berechnen.

Soll Kriterium	Kurzbezeichnung	durchschnittlicher Zielerfüllungsfaktorwert
1	Kompatibilität mit Naemon	1,17
2	Weiterverwendung individueller Check-Plugins	2,33
3	Supportunterstützung	1,83
4	Kosten für Lizenzierung und Support	1,50
5	Verwaltung der Agenten	0,83
6	Interkompatibilität	0,67

Tabelle 17: Durchschnittliche Zielerfüllungsfaktorwerte

Drei durchschnittliche Zielerfüllungsfaktorwerte stechen hierbei nach oben bzw. unten heraus.

- Hoher durchschnittlicher Zielerfüllungsfaktorwert von 2,33 bei Soll Kriterium 2 - Weiterverwendung individueller Check-Plugins

  Die Unterstützung von (individuellen) Monitoring Check Plugins nach den Monitoring Plugins Development Guidelines [30] bzw. der dazu kompatiblen Nagios Plugins Development Guidelines [45] ist von den meisten Monitoring Agenten Lösungen gegeben (Centreon, NCPA, openITCOCKPIT und SNCleint+). Lediglich checkmk und Icinga unterstützen diese nicht.

- Niedriger durchschnittlicher Zielerfüllungsfaktorwert von 0,83 bei Soll Kriterium 5 - Verwaltung der Agenten

  Die untersuchten Lösungen bieten entweder keine (zentrale) Verwaltungsmöglichkeiten der Monitoring Agenten (Zielerfüllungsfaktor 0) oder lediglich in einem Umfang ohne nennenswerten Mehrwert für die ASFINAG Umgebung (Zielerfüllungsfaktor 1).

  In der Gesamtheit betrachtet sind die Aktivitäten der Hersteller bezüglich zentraler Agentenverwaltungsmöglichkeiten nur sehr rudimentär ausgeprägt und wenig dazu geeignet den laufenden Pflegeaufwand für die Agenten deutlich zu verringern.

  Lediglich checkmk hat sich dieser Thematik intensiver angenommen und bietet hierbei Mehrwerte bei der Konfigurationsverwaltung der Agenden. Wobei man auch bei checkmk zusätzlich eine Drittlösung für Software-Verteilung benötigt.

- Niedriger durchschnittlicher Zielerfüllungsfaktorwert von 0,67 bei Soll Kriterium 6 - Interkompatibilität

  Da lassen sich zwei konträre Strategien erkennen:

  Einerseits die Strategie eines Monitoring Agenten, welcher grundsätzlich für die Verwendung von beliebigen Monitoring Backend Lösungen ausgerichtet ist. Wie der abzulösende NSClient++, dessen Centreon Derivat oder auch SNClient+.

  Andererseits die Strategie, dass ein Monitoring Agent lediglich für die eigene Monitoring Backend Lösung kompatibel ist. Sei dies durch technische Umsetzungen wie bei checkmk und Icinga oder durch Lizenzbestimmungen wie bei NCPA.

  Dieses Verhalten ist wohl auch dem Umstand geschuldet, dass kein Hersteller einen Support samt SLA für die eigenen Windows Monitoring Agentenlösung anbietet bzw. anbieten will. Die Lizenzierung erfolgt bei fast allen untersuchten Herstellern ausschließlich über die Monitoring Backend Lösung (und die Verwendung des Monitoring Agenten ist nicht zusätzlich zu lizenzieren) oder im

Fall von Hersteller Consol ausschließlich über Dienstleistungen (gänzlich ohne Lizenzierungskosten).

Die Kompatibilität zu Lösungen mit anderen Monitoringansätzen wie metrikbasiertes Monitoring oder gar die Unterstützung von Schnittstellen zu Standardisierungsansätzen wie Open Telemetry ist von nahezu keiner Lösung gegeben. Lediglich SNClient+ bietet mit seinem integrierten Prometheus-Stack eine Schnittstelle zu metrikbasierten Monitoring Backen Lösungen.

Weiters fallen die Bewertungen von Soll Kriterium 1 – Kompatibilität mit Naemon – auf. Denn es gibt (bis auf eine Ausnahme: openITCOCKPIT) lediglich Zielerfüllungsfaktor 0 oder 3. Somit gar nicht kompatibel mit Naemon oder voll kompatibel mit Naemon. Es hat sich gezeigt, dass sich die Hersteller meist dazu entschlossen haben, die Agenten Eigenentwicklung proprietär durchzuführen und (bewusst) keine Kompatibilität mit anderen Monitoring Backend Lösungen anstreben. Die Hälfte der untersuchten Monitoring Agenten Lösungen sind nicht mit dem Naemon Monitoring Backend kompatibel. Was eine zusätzliche (Teil)Ablöse des bestehenden Monitoring Backends Naemon notwendig macht.

5.7.2 Architekturgrundmuster

Bei der Untersuchung der Alternativen konnten zwei Architekturgrundmuster angetroffen werden.

Architektur Grundmuster 1:

Der abzulösende NSClient++ entspricht diesem Grundmuster. Der daraus entstandene Centreon-Fork daher ebenfalls. Auch SNClient+, mit dem Ziel eine 1-zu-1 Ablöse des NSClient++ zu bieten, kann in diesem Grundmuster betrieben werden. Weiters können auch die Alternativen openITCOCKPIT und NCPA mit diesem Architektur Grundmuster betrieben werden.

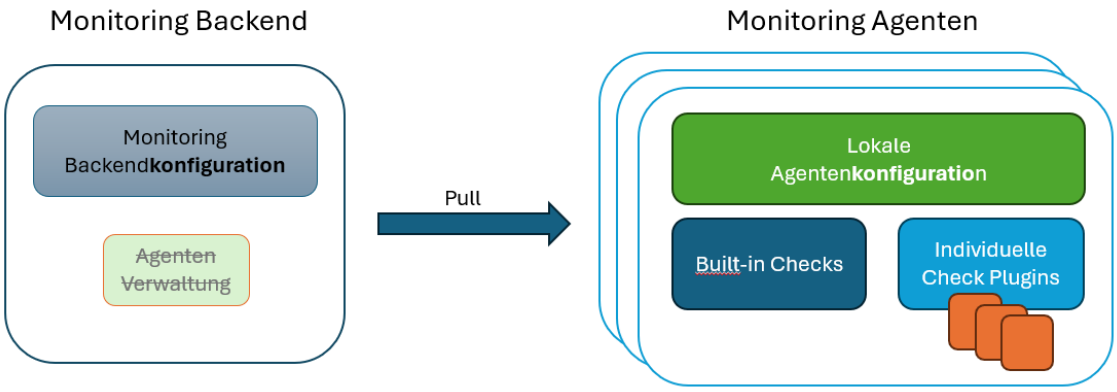

Abbildung 1: Architektur Grundmuster 1

Alle Alternativen, die in diesem Architekturmuster betrieben werden (können), vereint die Eigenschaft, dass diese keine (zentrale) Agentenverwaltung anbieten.

Architektur Grundmuster 2:

Die Abbildung 2 zeigt das Architekturmuster von checkmk sowie Icinga. Wobei die Agenten Verwaltungsmöglichkeiten bei Icinga nur lediglich rudimentär ausfallen.

Die Alternativen openITCOCKPIT und NCPA bieten die Möglichkeit den jeweiligen Agenten in diesem Architekturmuster zu betreiben – jedoch ohne Agenten Verwaltung/Registrierung.

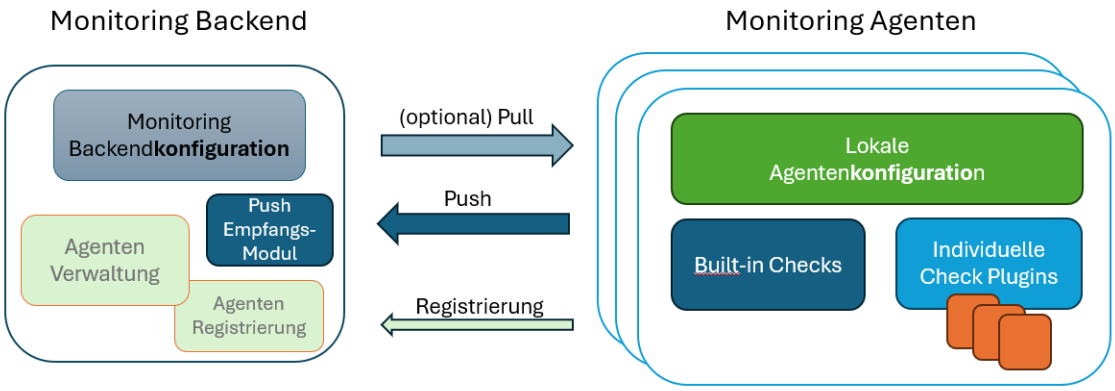

Abbildung 2: Architektur Grundmuster 2

Bei diesem Architekturmuster ist bei allen vier Alternativen (checkmk, Icinga, openITCOCKPIT und NCPA/NRDP) das dazu notwendige „Push Empfangs-Modul“ auf Monitoring Backend Seite nicht mit Naemon kompatibel, sondern nur mit dem jeweils eigenen Monitoring Backend.

Entweder in der technischen Implementierung (checkmk, Icinga, openITCOCKPIT) oder rechtlich über die gewählte open-source Lizenzierung untersagt (NRDP von Nagios Enterprises)

Bei checkmk und Icinga ist die (optionale) Pull Variante mit deren jeweiligen Monitoring Agenten ebenfalls nicht mit einem Naemon Backen kompatibel. Bei checkmk wegen der technisch unterschiedlichen Art der Ermittlung der Zustände (verteilt vs. zentral; siehe Kapitel 2.3). Bei Icinga ist die Web-API der Pull-Schnittstelle des Agenten als nicht für Drittsysteme vorgesehen deklariert und kann sich jederzeit ändern oder auch eingestellt werden.

Dies hat zur Folge, dass man für das Monitoring der Windows Umgebung mittels checkmk-Agent oder Icinga-Agent auch das jeweilige Backend System des Agenten-Herstellers mitverwenden muss, wenn man solch einen Windows Monitoring Agenten in diesem Architektur Grundmuster betreiben will.

Entweder muss dann die gesamte Monitoring Umgebung auf eine andere Backend Lösung migriert werden oder für das Monitoring der Windows Umgebung muss eine zweite Backend Lösungen parallel zum vorhandenen Monitoring Backend System betrieben werden. Sowie über eine Schnittstelle das zweite Monitoring Backen System (für das Monitoring der Windows Umgebung) an das bestehende Umbrella Backend Monitoring angebunden werden.

Dieses Architekturmuster mit push-Ansatz (und Registrierung bei einem zentralen Verwaltungsserver) ist bei Agenten-basierten IT-Lösungen dominant. Damit lässt sich auch das Backend-System in der Cloud bereitstellen bzw. als SaaS-Lösung (Software as a Service) betreiben. Was von den Herstellern checkmk als auch Icinga (über einen Partner) angeboten wird.

5.7.3 Aufwände und Kosten

Die Aufwände für die Ablöse von NSClient++ als Windows Monitoring Agent (Transitionsaufwand) sind neben den Lizenzierungskosten der größte Kostenfaktor.

Die Transitionsaufwände entstehen – je nach Alternative und Variante – durch folgende Tätigkeiten:

- Anpassung/Neuerstellung der lokalen Agentenkonfigurationen
- Anpassung/Neuerstellung der individuellen Check Plugins
- Aufwand für Dritt-Softwareverteilungslösung (Agenten-Software, individuelle Check-Plugins, lokale Agentenkonfiguration)
- Anpassung der bestehenden Naemon Backend Konfigurationen

- Betrieb einer zweiten Backend Lösung für Monitoring der Windows Umgebung samt Schnittstelle zu bestehendem Umbrella Monitoring System
- Komplettablöse der bestehenden Monitoring Lösung samt bestehendem Monitoring Backend

In der Bewertung zu Soll Kriterium 4 (Kosten für Lizenzierung und Support) sind die (grob abgeschätzten) Transitionsaufwände mit eingeflossen.

Bei der Alternative openITCOCKPIT, für die es mehrere technische Umsetzungsvarianten gibt, wurde die kostengünstigere Variante für die Bewertung herangezogen (Variante 2; Eigenentwicklung und Pflege eines eigenen Naemon Backend-Plugins für pull-Abfragen des Agenten; siehe Kapitel 5.4.2)

Auffällig ist, dass bei denjenigen Herstellern, die die Verwendung von deren Windows Monitoring Agenten an die eigene Monitoring Backend Lösung einschränken (checkmk, Icinga, NCPA) bei den Kosten am schlechtesten abschneiden und bei Soll Kriterium 4 (Kosten für Lizenzierung und Support) lediglich den Zielerfüllungsfaktor 0 (checkmk und Icinga) bzw. Zielerfüllungsfaktor 1 (NCPA) erreichen. Dies resultiert daraus, dass einerseits zusätzlich eine Migration des Monitoring-Backends erforderlich ist und damit einhergehende hohen Transitionsaufwänden/-kosten verursacht sowie zusätzlich hohe Lizenzierungskosten anfallen.

Wohingegen Alternativen, die nicht auf eine Monitoring Backend Lösungen eingeschränkt sind und bei den individuellen Check-Plugins eine offene Spezifikation setzen, wesentlich kostengünstiger sind. Die geringeren Kosten ergeben sich aus weniger Transitionsaufwänden als auch lizenzkostenfreie Nutzung des Agenten sowie keiner Notwendigkeit weitere Lizenzkosten für ein bestimmtes Monitoring Backend bezahlen zu müssen. Was sich in der Bewertung des Soll Kriterium 4 (Kosten für Lizenzierung und Support) im Zielerfüllungsfaktor 2 (openITCOCKPIT) bzw. Zielerfüllungsfaktor 3 (Centreon NSClient++ und SNClient+) widerspiegelt.

Ein Vendor Lock-In Effekt bei den Herstellern checkmk, Icinga und Nagios Enterprises (NCPA) ist erkennbar. Einerseits ist die Kompatibilität eingeschränkt bis gar nicht gegeben und anderseits sind die (Gesamt)Kosten deutlich höher.

5.8 Durchführung der Nutzwertanalyse

Die Bewertungen der Muss-Kriterien sowie der Soll-Kriterien (Zielerfüllungsfaktoren) der in den vorangegangenen Kapiteln untersuchten alternativen Windows Monitoring Agenten Produkte werden in die Nutzwertberechnungstabelle (siehe Kapitel 4.6) eingesetzt.

Über die Multiplikation der Bewertungsfaktoren (siehe Kapitel 4.3) mit den Zielerfüllungsfaktoren (siehe Kapitel 5.7) kann für jedes Soll-Kriterium ein dazugehöriger Nutzwert abgebildet werden.

Über die Aufsummierung der Einzelnutzwerte erhält man somit für jede Alternative einen Gesamtnutzwert.

			Centreon NSClient++		checkmk		NRDP (NCPA)	
Muss Kriterium	**Kurzbezeichnung**		**Zielerfüllung**		**Zielerfüllung**		**Zielerfüllung**	
1	Weiterentwicklung des Agenten		nicht erfüllt	-	erfüllt	-	erfüllt	-
2	Unterstützung individueller Checks		erfüllt	-	erfüllt	-	erfüllt	-
Soll Kriterium	**Kurzbezeichnung**	**Bewertungsfaktor**	**Zielerfüllung**	**Nutzwert**	**Zielerfüllung**	**Nutzwert**	**Zielerfüllung**	**Nutzwert**
1	Kompatibilität mit Naemon	0,25	3	0,75	0	0	0	0
2	Weiterverwendung individueller Check-Plugins	0,25	3	0,75	1	0,25	3	0,75
3	Supportunterstützung	0,15	1	0,15	2	0,3	2	0,3
4	Kosten für Lizenzierung und Support	0,25	3	0,75	0	0	1	0,25
5	Verwaltung der Agenten	0,05	1	0,05	2	0,1	0	0
6	Interkompatibilität	0,05	1	0,05	0	0	0	0
	Gesamt	**1,00**	ausgeschieden	**~~2,5~~**		**0,65**		**1,3**

Tabelle 18: Nutzwertberechnung Teil A

			openITCOCKPIT		Icinga		SNClient+	
Muss Kriterium	**Kurzbezeichnung**		**Zielerfüllung**		**Zielerfüllung**		**Zielerfüllung**	
1	Weiterentwicklung des Agenten		erfüllt	-	erfüllt	-	erfüllt	-
2	Unterstützung individueller Checks		erfüllt	-	erfüllt	-	erfüllt	-
Soll Kriterium	**Kurzbezeichnung**	**Bewertungsfaktor**	**Zielerfüllung**	**Nutzwert**	**Zielerfüllung**	**Nutzwert**	**Zielerfüllung**	**Nutzwert**
1	Kompatibilität mit Naemon	0,25	1	0,25	0	0	3	0,75
2	Weiterverwendung individueller Check-Plugins	0,25	3	0,75	1	0,25	3	0,75
3	Supportunterstützung	0,15	2	0,3	2	0,3	2	0,3
4	Kosten für Lizenzierung und Support	0,25	2	0,5	0	0	3	0,75
5	Verwaltung der Agenten	0,05	0	0	1	0,05	1	0,05
6	Interkompatibilität	0,05	1	0,05	0	0	2	0,1
	Gesamt	**1,00**		**1,85**		**0,6**		**2,7**

Tabelle 19: Nutzwertberechnung Teil B

Die Alternative Centreon NSClient++ wird wegen nicht aktiver Weiterentwicklung der eigentlichen Agentensoftware ausgeschieden.

Den höchsten Gesamtnutzwert erzielt mit großem Abstand die Alternative SNClient+. Gefolgt von der Alternative openITCOCKPIT.

Die Unterschiede beim Nutzen zwischen dem zweitplatzierten openITCOCKPIT und der der erstgereihten Lösung SNClient+ liegen im Wesentlichen bei der besseren Kompatibilität mit Naemon (Anpassungsbedarf der vorhandene Backendkonfiguration) sowie damit verbundenen Kosten.

Die restlichen Alternativen (checkmk, NRDP/NCPA, Icinga) haben einen deutlich geringeren Nutzwert, da wegen Inkompatibilitäten zum Naemon Backend zusätzlich eine Migration auf Monitoring-Backendseite notwendig wird und diese neben zusätzlichen

Lizenzierungskosten auch sehr hohe Transitionsaufwände mit sich ziehen (siehe dazu auch Kapitel 5.7.3)

Von diesen drei Lösungen hat NRDP/NCPA einen höheren Nutzwert gegenüber checkmk und Icinga, da bei dieser Lösung zumindest die vorhandenen individuellen Check-Plugins ohne Anpassungsbedarf weiterverwendet werden können.

Die Reihung der Alternativen hat sich durch die Bewertungsfaktoren der Soll-Kriterien nicht verändert. Jedoch haben sich die Abstände zwischen den beiden Erstplatzierten (SNClient+ und openITCOCKPIT) zu den drei anderen nicht ausgeschiedenen Alternativen (Icinga, checkmk und NCPA) vergrößert. Dies deshalb, weil der Schwerpunkt der Gewichtung auf wenig Transitionsaufwand (Kompatibilität mit bestehenden Naemon Monitoring Backend, Weiterverwendung bestehender individueller Check Plugins) sowie den Kosten liegt.

Dadurch lässt sich begründen, weshalb die Nutzwerte - zwischen niedrigstem Nutzwert von 0,6 und höchstem Nutzwert von 2,7 liegt ein Faktor von 4,5 - recht weit auseinanderliegen.

6. Bewertung der Hypothese und Beantwortung der Forschungsfrage

Die Untersuchung der Windows Monitoring Agenten Alternativen anhand der durchgeführten Nutzwertanalyse hat einen eindeutigen Erstgereihten mit SNClient+ sowie auch einen klaren Zweitgereihten mit openITCKOCKPIT ergeben (siehe Kapitel 5.8).

Diese beiden Lösungen sind am besten dafür geeignet, um den vorhandenen Windows Monitoring Agenten NSClient++ abzulösen. Mit diesen beiden Lösungen ist am wenigsten Aufwand sowie Kosten für die Ablöse verbunden. Beide Lösungen sind lizenzkostenfrei nutzbar, die vorhandenen individuellen Check-Plugins können ohne Anpassungen genutzt werden. Bei SNClient+ kann sogar die vorhandene Monitoring Backend Konfiguration kann ohne Anpassungen weiterverwendet werden.

Die anderen untersuchten Alternativen bringen neben zusätzlichen Lizenzkosten auch hohe Transitionskosten wegen notwendiger (Teil)Ablöse des vorhandenen Monitoring Backends samt neu zu erstellender Backend-Konfiguration sowie mitunter Neuentwicklung aller individueller Check-Plugins mit sich.

Siehe dazu die Agenten-Untersuchungen in den Kapiteln 5.2.2, 5.3.2, 5.4.2, 5.5.2, 5.6.2, sowie der Kostenvergleich über Soll-Kriterium 4 in Tabelle 16 und die Ausführungen über Aufwände und Kosten in Kapitel 5.7.3.

Die Auswahl der Alternativen für die Nutzwertanalyse kommt aus einer zuvor durchgeführten Markterkundung (siehe Kapitel 1.3). Die Markterkundung hat ergeben, dass es keine allzu große Auswahl gibt und man mitunter nicht an einer (Teil)Ablöse des bestehenden Monitoring Backends herumkommt. Daher wurden die Kriterien der Nutzwertanalyse entsprechend definiert (siehe Kapitel 4.2) und in der Gewichtung so festgelegt (siehe Kapitel 4.3), dass in der Gesamtbetrachtung eine Bestlösung ermittelt wird (siehe Kapitel 5.8).

Wenn die Gesamtaufwände bzw. Gesamtkosten trotz (Teil)Ablöse des bestehenden Monitoringbackends günstiger ausfallen als eine teure „eins-zu-eins“-Agentenablöse, dann soll die sinnvollere Alternative gefunden werden – im Sinne eines besseren Aufwandes/Kosten zu Nutzen Verhältnisses.

Die durchgeführte Nutzwertanalyse hat jedoch gezeigt, dass die „inkompatibleren“ Lösungen, bei denen eine (Teil)Ablöse des vorhandenen Monitoring Backends

notwendig wird, höhere Lizenzkosten als auch höhere Transistionskosten generieren. Wohingegen die „kompatibleren“ Lösungen sowohl geringe Lizenz- als auch geringe Transitionskosten aufweisen. Im Regelfall gehen höherer Lizenzkosten einher mit mehr Nutzungsmöglichkeiten oder geringerem Aufwand was Transitions- als auch laufenden Aufwand betrifft. Bei der Untersuchung der Windows Agenten Alternativen hat sich allerdings mitunter gegenteiliges gezeigt mit Vendor Lock-In Strategien und negativen Auswirkungen bezüglich Transitionsaufwände wie auch Gesamtkosten (siehe Kapitel 5.7.3.).

Da die derzeitige Verwaltung des Windows Monitoring Agenten NSClient++ sehr aufwendig ist, wurde ein weiterer (wenn auch etwas geringer gewichteter) Schwerpunkt auf die Möglichkeiten zur (zentralen) Verwaltung von Monitoring Agenten Alternativen gelegt. Bei einer Lösung mit gut nutzbaren zentralen Verwaltungsmöglichkeiten wäre eine Aufwands- und damit Kosteneinsparung im Verglich zur derzeitigen Situation möglich (siehe Kapitel 4.5.5.).

Die Untersuchung der Alternativen hat ergeben, dass die meisten Lösungen keinen großen Wert auf zentrale Agentenverwaltungsmöglichkeiten legen. Einzig checkmk zeigt dahingehend Engagement. Wobei dennoch auch bei checkmk eine Drittlösung für Softwareverteilung weiterhin notwendig ist. Weitere Hersteller zeigen erste Schritte in die Richtung zentraler Verwaltungsmöglichkeiten der Agenten, jedoch sind diese Aktivitäten nur sehr rudimentär und wenig dazu geeignet den laufenden Pflegeaufwand für die Agenten deutlich zu verringern. Siehe Kapitel 5.7.1 (Niedriger durchschnittlicher Zielerfüllungsfaktorwert von 0,83 bei Soll Kriterium 5 - Verwaltung der Agenten).

Mituntersucht (mit geringerer Gewichtung) wurde weiters die Interkompatibilität von Monitoring Agenten mit Standards wie OpenMetrics bzw. OpenTelemetry aus dem aufstrebenden Observability Umfeld, was als Weiterentwicklung des klassischen Monitorings angesehen werden kann. Einzig SNClient+ zeigt mit dem im Agenten eingebauten Prometheus-Stack (samt Exporter) Initiative in diesem Bereich. Alle weiteren Alternativen lassen keinerlei Aktivitäten in diese Richtung erkennen. Siehe Kapitel 5.7.1 (Niedriger durchschnittlicher Zielerfüllungsfaktorwert von 0,67 bei Soll Kriterium 6 - Interkompatibilität).

6.1 Bewertung der Hypothese

Die durchgeführte Nutzwertanalyse hat die Hypothese

> „Das Windows Monitoring Agent Produkt openITcockpit von der it-novum GmbH ist am besten geeignet um den bestehenden Windows Monitoring Agenten NSClient++ bei der ASFINAG IT abzulösen.“ (siehe Kapitel 1.6)

widerlegt und somit falsifiziert.

Denn laut durchgeführter Nutzwertanalyse ist die Alternative SNClient+ die am besten dafür geeignete Windows Monitoring Agentenlösung.

Die Hypothese baut auf dem Ergebnis der Markterkundung der Cancom auf (siehe Kapitel 1.3). Diese wurde im Sommer 2023 durchgeführt. Zu diesem Zeitpunkt befand sich das Windows Monitoring Agenten Produkt SNClient+ von Consol noch in einem frühen Entwicklungsstadium und war damals noch mit einem geringen Funktionsumfang ausgestattet und daher in der Marktanalyse nicht bestgereiht. Die Agentenlösung SNClient++ hat sich im weiteren zeitlichen Verlauf bis zum Jahresende 2023 im Sinne der ASFIANAG Anforderungen positiv weiterentwickelt. Die in der damaligen Marktanalyse als potenziell bestgeeignete Lösung hervorgegangene openITCOCKPIT ist in der durchgeführten Nutzwertanalyse als Alternative mit dem zweithöchsten Nutzwert bewertet worden.

6.2 Beantwortung der Forschungsfrage

Die Forschungsfrage (siehe Kapitel 1.5) lässt sich somit folgendermaßen beantworten:

Das Windows Monitoring Agenten Produkt SNClient+ von Consol ist am besten geeignet, um den bisher verwendeten Windows Monitoring Agenten NSClient++ bei der ASFINAG IT abzulösen.

7. Zusammenfassung, Limitationen & Ausblick

Um herauszufinden, mit welchem Windows Monitoring Agent Produkt das derzeit verwendete Produkt NSClient++ in der ASFINAG IT am besten abgelöst werden kann (siehe Kapitel 1.7 und Kapitel 1.4) wurde eine Nutzwertanalyse durchgeführt.

Die durchgeführte Nutzwertanalyse hat einen eindeutigen Erstgereihten mit SNClient+ ergeben (siehe Kapitel 5.8).

7.1 Limitationen

Obwohl jede der untersuchten Monitoring (Backend) Lösungen mit dem Windows Monitoring Agenten NSClient++ nutzbar ist, hat sich gezeigt, dass die meisten Hersteller die Entwicklung eines eigenen Monitoring Agenten bevorzugt haben. Die jeweiligen Agenten-Eigenlösung sind jedoch von einigen Herstellern bewusst so ausgeführt worden, dass Monitoring Backends von anderen Herstellern technisch und/oder juristisch nicht nutzbar sind. Dies betrifft die Agenten Produkte der Hersteller checkmk, Icinga und Nagios Enterprises mit NCPA (siehe Kapitel 5.7.1).

Weiters wird auch die Entwicklung von individuellen Check-Plugins bei zwei der untersuchten Agentenlösung proprietär ausgeführt (checkmk und Icinga). Was zur Folge hat, dass bereits vorhandene individuelle für NSClient++ entwickelte Check-Plugins nicht weiterverwendet werden können, sondern neu entwickelt werden müssen. Jedoch hat sich auch gezeigt, dass die offenen Spezifikationen zu Monitoring Check Plugins nach den Monitoring Plugins Development Guidelines [30] bei den untersuchten Monitoring Agenten Alternativen weit verbreitet ist und von Centreon, NCPA, openITCOCKPIT und SNClient+ unterstützt werden (siehe Kapitel 5.7.1 bzw. Untersuchungen der Alternativen auf Soll Kriterium 2 - Weiterverwendung individueller Check-Plugins).

Die Möglichkeiten zur (zentralen) Verwaltung der untersuchten Windows Monitoring Agenten ist gar nicht bis lediglich sehr rudimentär vorhanden. Einzige Ausnahme ist checkmk, die sich dieser Thematik ernsthaft angenommen haben (siehe Kapitel 5.7.1).

Interkompatibilität mit Standards aus dem Observability Umfeld ist bei den untersuchten Windows Monitoring Agenten – mit Ausnahme von SNClient+ mit Metriken über Prometheus-Exporter – nicht gegeben (siehe Kapitel 5.7.1).

7.2 Ausblick

Die Untersuchung basiert im Wesentlichen aus Papierarbeit wie Internetrecherche sowie Anfragen an Hersteller und deren Partner.

Als nächsten Schritt macht es Sinn für die erstgereihte Alternative SNClient+ – sowie eventuell auch für die zweitgereihte Alternative openITCOCKPIT – einen sogenannten Proof of Concept (kurz: POC) aufzusetzen. Unter einem POC versteht man einen Nachweis dafür, dass sich ein theoretisch erarbeitetes Vorhaben auch in der Praxis umsetzen lässt. Man testest somit eine Lösungsvariante in der Praxis, indem eine paar Agenteninstallationen auf verschiedenen Windowssystemen durchgeführt und darauf Testfälle abgearbeitet werden. Sinnvollerweise berücksichtigt man beim Testen auf die in der Nutzwertanalyse festgehaltenen Soll-Kriterien (z.B. Kompatibilität mit Naemon testen, funktionieren die vorhandenen Check-Plugins tatsächlich ohne Anpassungen, ist die Naemon Backendkonfiguration mit der Agentenalternative tatsächlich ohne Anpassung weiterhin nutzbar).

Bei den Möglichkeiten der zentralen Verwaltung von Monitoring Agenten gibt es noch sehr viel Potential, um Mehrwerte für Kunden generieren zu können. Checkmk wird den eingeschlagenen Weg wohl fortsetzen und ausbauen. Möglicherweise folgen weitere Hersteller und implementieren Unterstützungen für die zentrale Agentenverwaltung in ihren jeweiligen Lösungen.

Bei der Interoperabilität haben sich die Agenten im klassischen Monitoringumfeld auseinanderentwickelt. In den letzten Jahren stattgefundene Monitoring Agenten wurden proprietär ausgeführt (checkmk, NCPA, Icinga).

Im Gegensatz dazu sind die meisten Hersteller von APM/Observability Lösungen den gegengesetzten Weg gegangen und haben über die Vernetzungsorganisationen OpenMetrics und OpenTelemetry u.a. eine Kommunikations-/Protokollstandardisierung zwischen Agenten und Backendsystemen geschaffen (siehe Kapitel 3.4)

Es wird interessant sein, wie die Hersteller von klassischen Monitoringsystemen Auf Backend- wie auch auf Agenten-Seite darauf reagieren werden, wenn zukünftig weitere OpenTelemetry kompatible Monitoring Agenten im Umlauf sein werden. Wenn etwa Microsoft eine OpenTelemetry Schnittstelle in das Windows Betriebssystem integriert

sowie individuelle Erweiterungen ermöglicht, dann wäre ein Agentless Monitoring bei Windows Systemen möglich und kein eigener Windows Monitoring Agent mehr notwendig (siehe Kapitel 3.4.2).

Das Monitoring Backend Naemon ist jedoch derzeit nicht kompatibel mit OpenTelemetry (siehe Kapitel 3.5). Möglicherweise wird sich dies ändern und Naemon als auch andere Monitoring Backend Lösungen OpenTelemetry zukünftig unterstützen. Damit wären die eigenen Monitoring Agenten Lösungen nicht weiter notwendig. Somit ist es denkbar, dass die Backends der klassischen Monitoringhersteller die OpenTelemetry implementieren werde und die eigene Entwicklung von Monitoring Agenden einstellen werden.

Andererseits ist es auch möglich, dass ein Ansatz eines universellen Monitoring Agenten anklang findet und ein Monitoring Agent nicht nur beliebige klassische Monitoring Backend Lösungen unterstützt (wie NSClient++ oder SNClient+), sondern auch weitere OpenTelemetry kompatible Monitoring Backend Lösungen. Von den untersuchten Alternativen bringt SNClient+ mit dem Prometheus-Stack eine Unterstützung von metrikbasierten Monioring Backends mit (siehe Kapitel 5.6.2 sowie Kapitel 5.7.1). Da OpenMetrics stark von Prometheus beeinflusst ist (siehe Kapitel 3.4.1) und OpenMetrics ein Teil von OpenTelemetry darstellt (siehe Kapitel 3.4.2), wäre eine technische Umsetzung in SNClient+ nicht besonders aufwändig, um Metriken per OpenTelemetry Schnittstelle bereitstellen zu können.

Es ist auch denkbar, dass weitere Monitoring Agenten Hersteller ihre Monitoring Agenten um eine OpenTelemetry Schnittstelle erweitern. Jedoch würde dies eine Strategieänderung von proprietär zu allgemein-kompatibel für die Hersteller checkmk, Nagios Enterprises und Icinga bezüglich ihrer Monitoring Agenten bedeuten.

Literaturverzeichnis

[1] „ConSol-Monitoring/omd“. ConSol Software GmbH - Monitoring Team, 18. Dezember 2023. Zugegriffen: 11. Januar 2024. [Online]. Verfügbar unter: https://github.com/ConSol-Monitoring/omd

[2] „Nagios Open Source | Nagios Open Source“. Zugegriffen: 11. Januar 2024. [Online]. Verfügbar unter: https://www.nagios.org/

[3] „Naemon - Monitoring Suite“. Zugegriffen: 11. Januar 2024. [Online]. Verfügbar unter: https://www.naemon.io/

[4] „Welcome - NSClient++“. Zugegriffen: 11. Januar 2024. [Online]. Verfügbar unter: https://nsclient.org/

[5] „Nagios-NRPE/share/protocol-nrpe.md at master · stockholmuniversity/Nagios-NRPE“, GitHub. Zugegriffen: 11. Januar 2024. [Online]. Verfügbar unter: https://github.com/stockholmuniversity/Nagios-NRPE/blob/master/share/protocol-nrpe.md

[6] „Magic Quadrant Research Methodology“, Gartner. Zugegriffen: 13. Januar 2024. [Online]. Verfügbar unter: https://www.gartner.com/en/research/methodologies/magic-quadrants-research

[7] „GitHub - centreon/centreon-nsclient-build: Source use to build the centreon NSClient agent“. Zugegriffen: 13. Januar 2024. [Online]. Verfügbar unter: https://github.com/centreon/centreon-nsclient-build

[8] „Infrastructure & Application Monitoring with Checkmk“, Checkmk. Zugegriffen: 13. Januar 2024. [Online]. Verfügbar unter: https://checkmk.com/

[9] „NagiosEnterprises/nrdp“. Nagios Enterprises, 19. November 2023. Zugegriffen: 13. Januar 2024. [Online]. Verfügbar unter: https://github.com/NagiosEnterprises/nrdp

[10] „openITCOCKPIT | Open Source Monitoring Configuration Interface“. Zugegriffen: 13. Januar 2024. [Online]. Verfügbar unter: https://openitcockpit.io/

[11] „Icinga » Monitor your entire Infrastructure with Icinga“, Icinga. Zugegriffen: 13. Januar 2024. [Online]. Verfügbar unter: https://icinga.com/

[12] „ConSol-Monitoring/snclient“. ConSol Software GmbH - Monitoring Team, 2. Januar 2024. Zugegriffen: 13. Januar 2024. [Online]. Verfügbar unter: https://github.com/ConSol-Monitoring/snclient

[13] „prometheus-community/windows_exporter“. Prometheus Monitoring Community, 11. Januar 2024. Zugegriffen: 11. Januar 2024. [Online]. Verfügbar unter: https://github.com/prometheus-community/windows_exporter

[14] P. Donko, J. R. Mühlbacher, und R. Hörmanseder, „Network-Monitoring im Netzwerk einer Tageszeitung“, 2008. [Online]. Verfügbar unter: https://api.semanticscholar.org/CorpusID:62659566

[15] H. Schwantner, „MonitoringheterogenerSystemeamBeispieldesIT-Systems im“DasTIETZ““. Hochschule Mittweida (FH) FachbereichElektro-und Informationstechnik, Mai 2014. Zugegriffen: 13. Januar 2024. [Online]. Verfügbar

unter: https://monami.hs-mittweida.de/frontdoor/deliver/index/docId/4468/file/DA_Holm_Schwantner_Monitoring.pdf

[16] „Zabbix :: The Enterprise-Class Open Source Network Monitoring Solution“. Zugegriffen: 13. Januar 2024. [Online]. Verfügbar unter: https://www.zabbix.com/index

[17] M. Mormul, „Konzepte zur Verbesserung des Monitoring in industriellen Cloud-Umgebungen“, 2022, doi: 10.18419/OPUS-12220.

[18] Prometheus, „Prometheus - Monitoring system & time series database“. Zugegriffen: 14. Januar 2024. [Online]. Verfügbar unter: https://prometheus.io/

[19] D. Urban, „Optimierung der Visualisierung eines Dashboards für das Microservice-Monitoring“. Universität Leipzig Wirtschaftswissenschaftliche Fakultät Wirtschaftsinformatik/ Softwareentwicklung für Wirtschaft & Verwaltung, 5. November 2021. [Online]. Verfügbar unter: https://nbn-resolving.org/urn:nbn:de:bsz:15-qucosa2-768239

[20] J. Livens, „Observability vs. monitoring: What’s the difference?“, Dynatrace news. Zugegriffen: 14. Januar 2024. [Online]. Verfügbar unter: https://www.dynatrace.com/news/blog/observability-vs-monitoring/

[21] „Observability vs. Monitoring: Understanding the Difference | StrongDM“. Zugegriffen: 14. Januar 2024. [Online]. Verfügbar unter: https://www.strongdm.com/blog/observability-vs-monitoring

[22] M. Graf, „Bedeutung von Telemetrie für den Software Development Life Cycle“, Bachelor Thesis. [Online]. Verfügbar unter: https://nbn-resolving.org/urn:nbn:de:bsz:840-opus4-1751

[23] „OpenMetrics/specification/OpenMetrics.md at main · OpenObservability/OpenMetrics“, GitHub. Zugegriffen: 14. Januar 2024. [Online]. Verfügbar unter: https://github.com/OpenObservability/OpenMetrics/blob/main/specification/OpenMetrics.md

[24] „OpenTelemetry“, OpenTelemetry. Zugegriffen: 14. Januar 2024. [Online]. Verfügbar unter: https://opentelemetry.io/

[25] „Cloud Native Computing Foundation“, CNCF. Zugegriffen: 14. Januar 2024. [Online]. Verfügbar unter: https://www.cncf.io/

[26] V. Weimert, „Optimierung der Geschäftsprozesse im Affiliate Marketing – Konzeptentwicklung zur Einführung eines CRM-Systems im Auftrag der OnMaCon GmbH“. Hochschule für angewandte Wissenschaften Hamburg, 2012. [Online]. Verfügbar unter: http://hdl.handle.net/20.500.12738/5926

[27] G. A. Winkelhofer, *Management- und Projekt-Methoden: ein Leitfaden für IT, Organisation und Unternehmensentwicklung*, 3., Vollst. überarb. Aufl. Berlin Heidelberg: Springer, 2005.

[28] S. Koch, *Einführung in das Management von Geschäftsprozessen: Six Sigma, Kaizen und TQM*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. doi: 10.1007/978-3-642-01121-4.

[29]„centreon/centreon-plugins“. Centreon, 7. Mai 2024. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://github.com/centreon/centreon-plugins

[30]„Monitoring Plugins Development Guidelines“. Zugegriffen: 11. Januar 2024. [Online]. Verfügbar unter: https://www.monitoring-plugins.org/doc/guidelines.html

[31]„Was ist die Open Monitoring Distribution“, Checkmk. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://checkmk.com/de/guides/open-monitoring-distribution

[32]„Download-Archiv für veraltete Checkmk-Versionen“, Checkmk. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://checkmk.com/de/download/archive

[33]„Werks“, Checkmk. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://checkmk.com/werks

[34]„Writing agent-based check plug-ins“, Checkmk Docs. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://docs.checkmk.com/latest/en/devel_check_plugins.html

[35]„comnetgmbh/check_cmkagent_active“. comNET, 5. November 2020. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://github.com/comnetgmbh/check_cmkagent_active

[36]„comnetgmbh/thruk-check-cmkagent“. comNET, 27. November 2018. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://github.com/comnetgmbh/thruk-check-cmkagent

[37]„Videos and Slides“, OSMC. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://osmc.de/archives/2018-2/videos-and-slides/

[38]„Monitoring Agent · NCPA“. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://www.nagios.org/ncpa/

[39]„Index of /downloads/ncpa3“. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://assets.nagios.com/downloads/ncpa3/

[40]„NagiosEnterprises/ncpa“. Nagios Enterprises, 8. Mai 2024. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://github.com/NagiosEnterprises/ncpa

[41]„Using Scripts / Plugins With NCPA“. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://nagiosenterprises.my.site.com/support/s/article/Using-Scripts-Plugins-With-NCPA-32a58c75

[42]„ncpa/client/check_ncpa.py at master · NagiosEnterprises/ncpa“, GitHub. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://github.com/NagiosEnterprises/ncpa/blob/master/client/check_ncpa.py

[43]„API Reference · Documentation · NCPA“. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://www.nagios.org/ncpa/help/3.0/api.html

[44]„Nagios Open Software License Version 1.3“. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://assets.nagios.com/licenses/nagios_open_software_license.txt

[45]„Development Guidelines · Nagios Plugins“. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://nagios-plugins.org/doc/guidelines.html

[46]„it-novum/openitcockpit-agent-go“. it-novum GmbH, 20. Oktober 2023. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://github.com/it-novum/openitcockpit-agent-go

[47]„Releases · it-novum/openitcockpit-agent-go“, GitHub. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://github.com/it-novum/openitcockpit-agent-go/releases

[48]„Define custom checks“, GitHub. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://github.com/it-novum/openitcockpit-agent-go/wiki/Define-custom-checks

[49]„Releases · Icinga/icinga-powershell-framework“, GitHub. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://github.com/Icinga/icinga-powershell-framework/releases

[50]„Icinga 2 2.14.2“, Chocolatey Software. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://community.chocolatey.org/packages/icinga2/

[51]„Custom Plugins - Icinga for Windows“. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://icinga.com/docs/icinga-for-windows/latest/doc/900-Developer-Guide/11-Custom-Plugins/

[52]„OMD Labs“, ConSol Monitoring. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://omd.consol.de/docs/omd/

[53]„Releases · ConSol-Monitoring/snclient“, GitHub. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://github.com/ConSol-Monitoring/snclient/releases

[54]„OSMC 2023 | Replacing NSClient++ for Windows Monitoring by Sven Nielein“, SlideShare. Zugegriffen: 12. Mai 2024. [Online]. Verfügbar unter: https://www.slideshare.net/NETWAYS/replacing-nsclient-for-windows-monitoring-by-sven-nielein

Tabellenverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

ITSM	IT Service Management
NCPA	Nagios Cross-Plattform Agent
NMS	Netzwerk Management/Monitoring System
NRDP	Nagios Remote Data Processor
NRPE	Nagios Remote Plugin Executor
NSCA	Nagios Service Check Acceptor
OMD	Open Monitoring Distribution
OTel	Open Telemetry
POC	Proof of Concept
SIEM	Security Information Event Management
SNClient+	Secure Naemon Client
SNMP	Simple Network Management Protocol