Beschreibung von Fehlern in heterogenen Telekommunikationsnetzwerken und Ansätze zur Automatisierung des Troubleshootings

Jörg Benze
Competence Center Enterprise Management Systems (CC EMS)
Controlware GmbH, Waldstraße 92, D-63128 Dietzenbach
joerg.benze@controlware.de

Abstract. Die Qualität des Netzwerkmanagements und des damit verbundenen Entstörungsmanagements hängt heute vom Wissen und der Erfahrung der Operator ab. Um ein effizientes Fehlermanagement in komplexen Netzwerken zu betreiben, sollte das Entstörungsmanagement zukünftig nicht mehr vom Wissensstand der Operator abhängen. Dies kann mit einer Automatisierung des Fehlermanagements erreicht werden.

1 Einleitung

Ein Netzwerkmodell ist die Basis für automatische Verfahren im Bereich des Netzwerkmanagements. Das hier dargestelle objektorientierte Netzwerkmodell, welches als Basis für die Fehlerbeschreibung dient, wird aus den Prinzipien der relationalen Modelle entwickelt, auf denen die heutigen Anwendungen zur Netzwerkdokumentation basieren. Für ein (vereinfachtes) Beispiel eines Netzwerkmodells werden zwei *Managed Network Entities* (MNE) und deren Verbindung in Abbildung 1 gezeigt. Das Netzwerkmodell besteht aus folgenden Netzwerkdokumentationsobjekten:

Node: Eine Node ist das Basisobjekt zur Beschreibung einer MNE. Beispielsweise wird ein Chassis einer Netzwerkkomponente als Node modelliert.
Card: Eine Card beschreibt eine Karte, die in ein Chassis eingeschoben wird.
Port: Ein Port beschreibt einen Anschluß auf einer Karte, die über vielfältige Arten von Ports verfügen kann.
Link: Ein Link beschreibt eine Verbindung zwischen zwei Ports, die eine physikalische Datenübertragung ermöglicht (z.B. Kupfer-Leitung, Licht-Wellen-Leiter, Richtfunk etc.).
Circuit: Ein Circuit beschreibt eine logische Verbindung, die auf einem Link oder einer anderen logischen Verbindung (Circuit) basiert.
Service: Ein Service beschreibt einen Dienst, den das Netzwerk für einen Nutzer erbringt.
Subscriber: Ein Subscriber ist ein Nutzer des Netzwerks, der Dienste des Netzwerks in Anspruch nimmt.

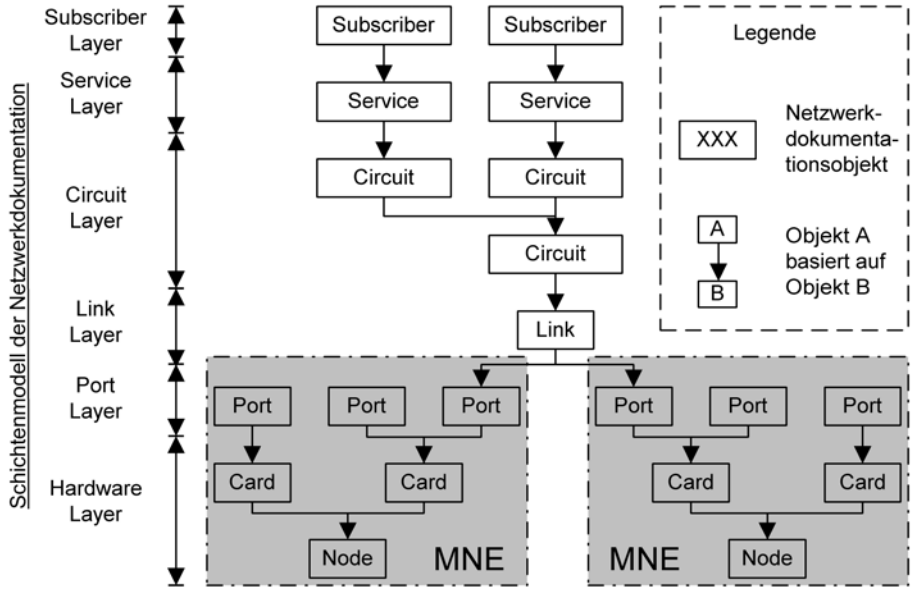


Abbildung 1: Modellierung von zwei *Managed Network Entities* (MNE) und der physikalischen und logischen Verbindungen dazwischen (vereinfacht).

2 Schichtenmodell der Netzwerkdokumentation

Betrachtet man den Aufbau des Netzwerkmodells in Abbildung 1, so lassen sich sechs Layer angeben, aus denen das *Schichtenmodell der Netzwerkdokumentation* entwickelt wird (siehe Abbildung 2). Hierbei haben die Layer folgende Funktion:

Hardware-Layer: Aufgabe des Hardware-Layers ist die Beschreibung des physikalischen Aufbaus der MNE des Netzwerks (außer den Ports).
Port-Layer: Aufgabe des Port-Layers ist die Beschreibung der physikalischen und logischen Anschlüsse der MNE. Der Port-Layer korrespondiert mit Teilen des OSI-1-Layers.
Link-Layer: Aufgabe des Link-Layers ist die Beschreibung der physikalischen Verbindungen zur Signalübertragung zwischen den Ports. Der Link-Layer korrespondiert mit Teilen des OSI-1-Layers.
Circuit-Layer: Aufgabe des Circuit-Layers ist eine Beschreibung der logischen Verbindungen, die auf einem Link betrieben werden. Circuits können ineinander verschachtels sein, d.h. ein Circuit kann auf einem anderen basieren. Der Circuit-Layer ist mit den OSI-Layern 2 bis 6 vergleichbar.
Service-Layer: Aufgabe des Service-Layers ist die technische Beschreibung eines Dienstes im Netz. Desweiteren enthält der Service-Layer Informationen hinsichtlich

der betriebswirtschaftlichen Daten des Dienstes. Die technische Seite des Service-Layers ist mit dem OSI-7-Layer vergleichbar.
Subscriber-Layer: Aufgabe des Subscriber-Layers ist die Beschreibung der Nutzer des Netzwerks, die die verschiedenen Services im Netzwerks verwenden. Der Subscriber-Layer enthält somit eine Auflistung aller Nutzer des Netzwerks.

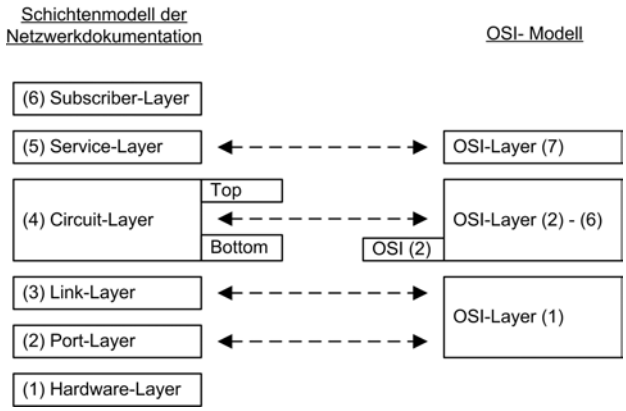


Abbildung 2: Schichtenmodell der Netzwerkdokumentation und Vergleich zum OSI-Modell

Vergleicht man das hier vorgestellte *Schichtenmodell der Netzwerkdokumentation* mit dem OSI-Modell so ist festzustellen, daß das *Schichtenmodell der Netzwerkdokumentation* das OSI-Modell nach oben und unten erweitert. Allerdings ist es in dem Bereich, der vom OSI-Modell mit abgedeckt wird, nicht so detailiert. Die Erweiterung nach unten erlaubt es, den physikalischen Aufbau der Netzwerkkomponenten darzustellen. Die Erweiterung nach oben ermöglicht es, den Nutzer eines Dienstes anzugeben.

3 Erweiterung des Netzwerkmodells zur Eventdarstellung

Zur Darstellung von Events im Netzwerkmodell und zur Planung bzw. Durchführung von Tests bzw. Troubleshooting Maßnahmen ist eine Erweiterung des Netzwerkmodells erforderlich. Es werden die Objekte *Test* und *Correct* eingeführt (siehe Abbildung 3), die mit den Netzwerkobjekten verbunden sind.

Test: Test dient zur Beschreibung von Testroutinen, die die Funktionalität der ihr zugeordneten Teilkomponente online im Netzwerk nachprüfen können.
Correct: Correct dient zur Beschreibung von Korrekturroutinen, die beim Auftreten eines Fehlers an der ihr zugeordneten Komponente Troubleshootingmaßnahmen online im Netz durchführen können.

Die *Test*- und *Correct*-Objekte für die jeweiligen Netzwerkkomponenten können zukünftig von den Herstellern für die Aufnahme des Produkts in die Netzwerkdo-

kumentation mitgeliefert werden, so wie die Hersteller heute die Private-MIB für ihre Produkte zur Aufnahme in das Netzwerkmanagementsystem mitliefern.

Occurence: *Occurence* dienen zur Beschreibung von Events im Netzwerkmodell. Hierzu wird für jedes Event, welches in die Netzwerkdokumentation aufgenommen werden soll, ein *Occurence* Objekt erstellt. Dieses wird mit dem jeweiligen Netzwerkdokumentationsobjekt verbunden, welches von dem Event betroffen ist.

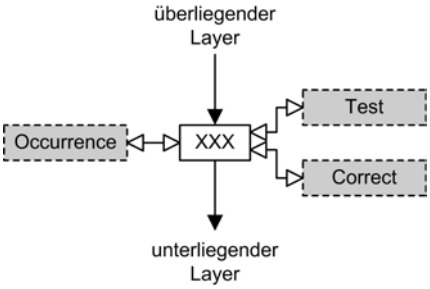


Abbildung 3: Erweiterung des Netzwerkmodells durch Objekte vom Typ *Test*, *Correct* und *Occurrence*, die mit den Netzwerkdokumentationsobjekten (vgl. Abbildung 1) verbunden sind.

4 Ansätze zur Automatisierung des Troubleshootings

Klassifikation

Die Abbildung der Events in das Netzwerkmodell ermöglichen eine automatische Klassifikation von Events in Kategorien, die dem *Schichtenmodell der Netzwerkdokumentation* entsprechen. Hieraus ergeben sich folgende Klassifikationskriterien:

- Hardware-Events
- Port-Events
- Link-Events
- Circuit-Events
- Service-Events
- Subscriber-Events

Hierbei ist anzumerken, daß ein Subscriber-Event von einem technischen Netzwerküberwachungssystem nicht generiert wird. Es könnte aber der Call eines Nutzers im Call-Center als Subscriber-Event in das Netzwerkmodell aufgenommen werden.

Event-Hierarchieordnung

Durch die Zuordnung der *Occurence*-Objekte zu den Netzwerkdokumentationsobjekten, die in der Hierarchieordnung des *Schichtenmodells der Netzwerkdokumentation* eingeordnet sind, läßt sich eine Hierarchieordnung der Occurence-Objekte (bzw. Events) ableiten (siehe Abbildung 4). Aus dieser wird ersichtlich, welche Services in Netzwerk gestört sind. Ferner erkennt man die Zusammenhänge, auf welche *Occur-*

ence die Störung der Services basiert und welche Beziehungen der *Occurence* untereinander bestehen.

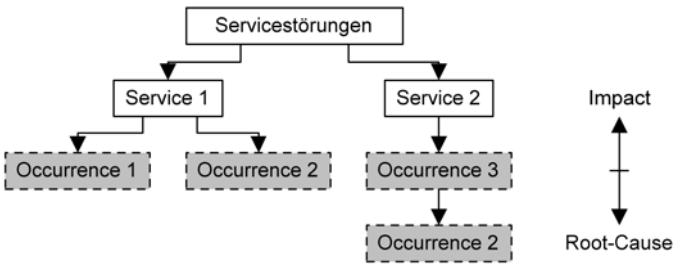


Abbildung 4: Beispiel für eine Event-Hierarchie

Die Hierarchieordnung der Events läßt sich für eine Root-Cause- und Impact-Analyse verwenden. Aus dem Beispiel in Abbildung 4 ist zu erkennen, daß die Störung des Service 1 ihren Grund in den *Occurence* 1 und 2 hat. *Occurence* 2 ist auch der Root-Cause für die Störung des Service 2. *Occurence* 3 basiert auf *Occurence* 2. Die Entstörung von *Occurence* 2 wird mit hoher Warscheinlichkeit *Occurence* 3 beseitigen.
Allgemein ist festzustellen, daß der Root-Cause für die Störung eines Services durch die Blattknoten seiner Eventhierarchie gebildet wird. Zur Impact-Analyse eines Occurence muß man ausgehend vom Netzwerkobjekt des Occurrence im Netzwerkmodell nach oben gehen, um alle Services zu ermitteln, die von dem Occurence betroffen sind.

Automatisches Troubleshooting

Auf Basis der Hierarchieordnung der *Occurence* (bzw. der Events), die für jeden gestörten Service erstellt wird, lassen sich Troubleshootingmaßnahmen mittels der *Test*- und *Correct*-Objekte im Netzwerkmodell durchführen. Abbildung 5 zeigt die Event-Hierarchie für einen Service.
Ausgehend von der Event-Hierarchie wird eine Prüfroutine aus den *Test*-Objekten des Netzwerkmodells erstellt, die die betroffenen Komponenten online im Netzwerk abfragt und eine detailierte Fehlerbeschreibung für diesen Service erstellt. Die Abfragen im Netz erfolgen in einer vorgegebenen Reihenfolge und beginnen immer mit den Blattobjekten der Hierarchie. Ausgehend von den Blattobjekten arbeitet man sich Schicht für Schicht bis zum Service nach oben.
Auf Basis des Ergebnisses der Prüfroutine wird mittels der *Correct*-Objekte eine Korrekturroutine erstellt. Der Ablauf des Troubleshootings (Korrekturroutine) in Abbildung 5 ergibt sich mit folgender beispielhafter Annahme. Die Occurence 1 und 4 basieren auf Events, die Folgeevents der Occurence 2, 3 und 5 sind. Dieser Sachverhalt wurde von der Prüfroutine ermittelt, so daß eine Entstörung der Occurence 1 und 4 nicht erforderlich ist. Aus diesem Grund nimmt die Korrekturroutine nur Troubleshootingmaßnahmen für die Occurence 2, 3 und 5 vor.

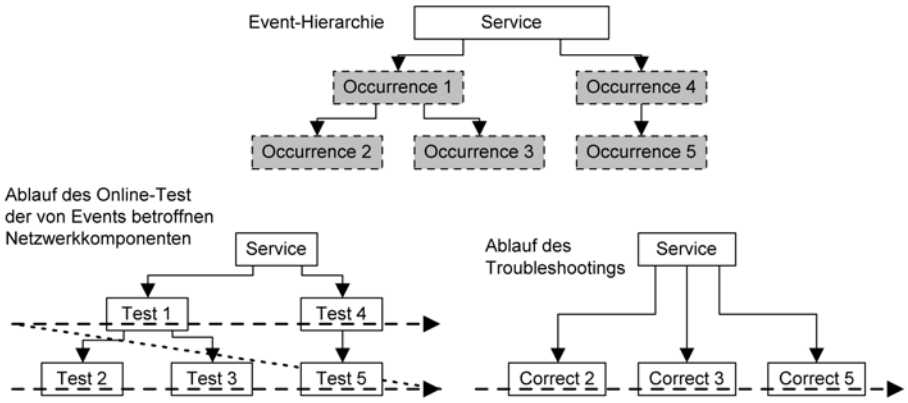


Abbildung 5: Beispiel für eine Event-Hierarchie, Ablauf der Fehlerdiagnose und Ablauf des Troubleshootings

5 Fazit

Im Gegensatz zu heutigen Verfahren bestehen bei dem vorgestellen Verfahren folgenden Vorteile:

- Erstellung und Durchführung von Troubleshootingmaßnahmen für beliebige Events. Gegenwärtige Systeme können nur eine vorher festgelegte Menge von Events verarbeiten, wobei pro Event der Operator händisch eine Entstörungsroutine erstellen muß.
- Das Verfahren ist robust gegenüber Änderungen in der Topologie und der Konfiguration des Netzwerks (vorausgesetzt das diese Veränderungen im Netzwerkmodell eingepflegt werden).
- Da *Test*- und *Correct*-Objekte für jedes Gerät vom Hersteller individuell mitgeliefert werden, kann dieses Verfahren in heterogene Netzwerkstrukuren eingesetzt werden.

References

1. Integriertes Management vernetzter Systeme, Heinz-Gerd Hegering * Sebastian Abeck * Bernhard Neumair, dpunkt-Verlag, Heidelberg 1999
2. Micromuse Netcool, http://www.micromuse.com, Produktdokumentationen und Schulungsunterlagen zur "Netcool Suite"
3. Smarts InCharge, http://www.smarts.com, Produktdokumentationen zu "InCharge"
4. Cramer Systems, http://www.cramer.com, Schulungsunterlagen zum "Certified Developer"