*Simulation in Produktion und Logistik*
*Entscheidungsunterstützung von der Planung bis zur Steuerung*
Wilhelm Dangelmaier, Christoph Laroque & Alexander Klaas (Hrsg.)
Paderborn, HNI-Verlagsschriftenreihe 2013

# Generierung eines Simulationsmodells auf Basis einer CMDB

***Generating a simulation model based on the Data of a CMDB***

Thorsten Kruse, Achim Schmidtmann, FH Dortmund, Dortmund (Germany),
kruse_thorsten@web.de, achim.schmidtmann@fh-dortmund.de

**Abstract:** The possibility to use simulations for analyzing local networks to prevent them from malfunctions and disruptions is rarely popular. Reasons for that are the high efforts for modeling the networks within a simulator and the determination of relevant data of the network. In this paper, therefore, an interface is developed that enables the possibility to retrieve the data of a network from a configuration management database and to create the appropriate model for the simulator afterwards. Furthermore, there is a detailed definition of the interface with the goal to build an interface that is flexible concerning future developments. Concluding, the functional capability of the interface is shown in an exemplary simulation.

## 1 Einführung

Die Grundlage für dieses Projekt bilden die fehlenden Möglichkeiten für Unternehmen präventive Maßnahmen für die Vermeidung von Ausfällen aber auch für die Planung von Erweiterungen und Umstrukturierungen ihrer IT-Infrastruktur. Grundsätzlich bietet sich hierfür die Simulation als geeignetes Mittel für die Analyse der Infrastruktur an, jedoch ist dies nur mit einem hohen Aufwand verbunden. Denn die erforderlichen Modelle und Komponenten in den verfügbaren Simulatoren müssen entwickelt bzw. entworfen werden. Hinzukommend müssen alle relevanten Daten des Netzwerkes erfasst und in die Simulation eingebracht werden. Für Unternehmen, die eine eigene Configuration Management Database, kurz CMDB, betreiben, ist es nahe liegend die Informationen aus dieser für die Simulation zu verwenden. Die Schwierigkeit besteht allerdings darin, die notwendigen Daten an den Simulator zu übergeben, da hierfür keine geeigneten Schnittstellen oder Verfahren existieren. Es ist somit erforderlich diese manuell aus der CMDB in den Simulator zu übertragen.

Das Ziel dieses Projektes ist es daher die oben dargestellte Problematik zu vermeiden, indem die Entwicklung einer geeigneten Schnittstelle dargestellt wird, die die Daten einer CMDB für einen Simulator zugänglich und nutzbar macht,

sodass mit diesem schnell und unkompliziert Simulationen durchgeführt werden können.

## 2 Anforderungen an die Schnittstelle

Die Aufgabe der Schnittstelle ist es die Daten der CMDB für den Simulator verfügbar zu machen. Wie in Abbildung 1 verdeutlicht ist, muss die CMDB zunächst ihre Daten in ein einheitliches Format konvertieren. Dieses ermöglicht es der Schnittstelle die Daten zu verstehen und diese in ein entsprechendes Modell für den Simulator um zu wandeln.

***Abbildung 1**:Ablauf der Modellerzeugung*

Der Schwerpunkt der zu entwickelnden Schnittstelle liegt dabei in der Definition eines einheitlichen Datenformats für den Austausch der Modelldaten. Die Umwandlung dieser Daten in ein Simulationsmodell muss dagegen für jeden Simulator gesondert vorgenommen werden, da diese kein standardisiertes Modellformat unterstützen.

Die Grundlage für die Entwicklung der Schnittstelle bildet die CMDB, welche die relevanten Daten für die Erzeugung eines Simulationsmodells beinhaltet. Diese ist ein Bestandteil des IT-Servicemanagements, wie es von der IT Infrastructure Library (kurz ITIL) und der ISO 20.000 beschrieben wird. Das Ziel des IT-Servicemanagement ist es IT-Leistungen bzw. Services für die Geschäftsprozesse eines Unternehmens bereitzustellen.

Um ein erfolgreiches IT-Servicemanagement umzusetzen haben sich dazu verschiedene Ansätze entwickelt. Sowohl ITIL als auch die ISO 20.000 stellen dabei einen prozessorientierten Ansatz für die Umsetzung des IT-Servicemanagements dar (Buchsein 2007).

In beiden Ansätzen wird die Simulation zwar nicht direkt als Instrument für Vermeidung bzw. Untersuchung von Störfällen vorgeschlagen, jedoch fordert die ISO 20.000 als auch ITIL proaktive Maßnahmen für die Vermeidung von Störungen (Kresse und Bause 2011; ISO/IEC 20000-1 2005).

Die CMDB ist in beiden Ansätzen zentraler Bestandteil des Configuration-Management-Prozess, der für die Pflege dieser verantwortlich ist. Die CMDB ist sowohl in ITIL als auch in der ISO 20.000 für die „Bereitstellung von aktuellen und historischen Informationen über die verfügbaren IT-Services, die damit verbundenen IT-Infrastruktur-Konfigurationselemente und deren Beziehungen untereinander“ (Wischki 2009) verantwortlich. Was für eine Vielzahl von Daten dabei für einen IT-

Service in der CMDB gespeichert werden muss, wird bei einer genauen Betrachtung deutlich. Ein IT-Service ist häufig eine bestimmte Anwendung, die von verschiedenen Clients genutzt wird. Die Anwendung selbst kann dabei z. B. auf einem Server laufen, der zusätzlich eine Datenbank benötigt, um die Daten der Anwendung speichern zu können. Die Datenbank läuft dabei meist ebenfalls auf einem oder mehreren Servern, welche ihre Daten möglicherweise in einem Speichersystem physikalisch hinterlegt haben. Um alle beteiligten Komponenten miteinander zu verbinden, wird zusätzlich ein Netzwerk benötigt, dass diese miteinander in Beziehung setzt und die Kommunikation ermöglicht. Außerdem benötigen alle Komponenten Strom, um überhaupt funktionieren zu können, wodurch Bedarf nach einem Stromnetz besteht. Abschließend ist zu berücksichtigen, dass alle genannten Bestandteile des IT-Service in einem bestimmten Raum und Gebäude untergebracht sind (Wischki 2009).

Neben den oben genannten Bestandteilen eines IT-Service muss darüber hinaus beachtet werden, dass jede Komponente aus unterschiedlich vielen Attributen und Parametern bestehen kann, wie z. B. ein Server, der aus verschiedenen Hardware-Komponenten besteht, die alle in der CMDB mit erfasst werden müssen. So wird deutlich, dass eine sehr detaillierte Erfassung aller Komponenten erforderlich ist und innerhalb der CMDB ein genaues Abbild der IT-Infrastruktur entsteht. Diese Genauigkeit ist allerdings keinesfalls unverhältnismäßig. Ein Ausfall der Klimaanlage des Rechenzentrums führt beispielsweise dazu, dass irgendwann die Server überhitzen, diese ausfallen oder sich selbstständig herunterfahren und der IT-Service für den Kunden nicht mehr zur Verfügung steht. Der Nutzen der CMDB wäre nun verfehlt, wenn das Problem in dieser erfasst werden soll und die relevante Komponente Klimaanlage nicht vorhanden ist, da sie nicht direkt am IT-Service beteiligt ist und deshalb nicht berücksichtigt wurde.

Wie die Betrachtung der CMDB zeigt, können umfangreiche Daten über die IT-Infrastruktur und deren Umfeld erfasst werden. Für die Schnittstelle bedeutet dies, dass sie möglichst alle Daten übertragen sollte, um eine möglichst breite Auswahl von unterschiedlichen Simulationsanalysen zu ermöglichen. Dabei müssen auch die unterschiedlichen Parameter der einzelnen Objekte der CMDB und auch die Vielzahl von möglichen Beziehungen unter den Objekten berücksichtigt werden. Hinzukommend muss die Schnittstelle auch zukunftssicher sein, um neue Komponenten übertragen zu können.

## 3 Definition der Schnittstelle

Grundsätzlich können die Daten einer CMDB in drei Bestandteile aufgegliedert werden – Objekte, Parameter, Beziehungen. Zu den Objekten zählen alle Komponenten die in der CMDB abgebildet werden können, wie z. B. Server, Router, aber auch Mitarbeiter, Serviceverträge und Peripheriegeräte. Die Beziehungen unter den Objekten ermöglichen es verschiedene Modelle in der CMDB anzulegen, wie z. B. Datennetze, Stromnetze o.ä.

Sowohl die Objekte in einer CMDB als auch die Beziehungen zwischen diesen, können Parameter besitzen, wie z. B. eine Übertragungs- oder Prozessorgeschwindigkeit. Für jede Simulation müssen die relevanten Parameter bereitgestellt werden, um diese überhaupt erst zu ermöglichen. Für die Definition der Schnittstelle bedeutet dies, dass alle drei Bestandteile berücksichtigt werden

müssen. In Algorithmus 1 ist die daraus resultierende Schnittstelle beispielhaft an einem Cluster dargestellt. Das Wurzelelement erlaubt es einen Namen als auch das Erstellungsdatum für das Modell anzugeben.

***Algorithmus 1:*** *Definition der Schnittstelle*

```
<model name="Ausfallsimulation" date="2012-07-03T09:00:00">
      <component id="10" name="Cluster" type="Cluster">
            <parameter name="Arbeitsspeicher">
                  <value>1024</value>
                  <unit>MB</unit>
            </parameter>

            <component name="Server 1" type="Server">
                  …
            </component>
      </component>
      <connection master="true">
            <from>
                  <id>10</id>
                  <name>Client1</name>
            </from>
            <to>
                  <id>9></id>
                  <name>Server1</name>
            </to>
            <direction>both</direction>
            <kind>network</kind>
            <parameter name="Übertragungsrate">
                  <value>1</value>
                  <unit>Gbit/s</unit>
            </parameter>
      </connection>
</model>
```

Das erste Unterelement der Schnittstelle stellt die Objekte dar, die in der CMDB abgebildet werden können. Eine Unterscheidung zwischen den unterschiedlichen Arten wird dabei durch das Attribut *Type* vorgenommen. Der Simulator kann dadurch die Komponenten unterscheiden und entsprechend in der Simulation berücksichtigen. Als weitere Attribute können eine ID und ein Name für das Objekt angegeben werden, wodurch eine Identifizierung durch den Anwender ermöglicht werden soll.

Als Unterelement der Komponenten können sowohl die für die Simulation erforderlichen Parameter als auch andere Komponenten angegeben werden. Für einen Computer können beispielsweise die einzelnen Hardware-Bestandteile als eigene Komponenten innerhalb der XML-Komponente angegeben werden. Abhängig von der Simulationsart und vom Detaillierungsgrad kann dies notwendig sein, wenn die Angabe von z. B. der Größe des Arbeitsspeichers oder der Prozessorgeschwindigkeit als Parameter nicht ausreichend ist und stattdessen detailliertere Informationen über die einzelnen Bestandteile erforderlich sind.

Die Parameter der Komponenten besitzen einen Namen, welcher ebenfalls zur Identifizierung sowohl für den Simulator als auch durch den Anwender erforderlich ist. Darüber hinaus kann ein entsprechender Wert und die zugehörige Maßeinheit angegeben werden. Hierdurch sollen Umrechnungsfehler zwischen der CMDB und dem Simulator vermieden werden. Für Parameter ohne Maßeinheit kann die Angabe dagegen entfallen.

Als letztes werden die Beziehungen zwischen den Objekten als separate XML-Komponenten in der Schnittstelle angegeben. Eine Zuordnung als Unterelement zu einem Objekt ist hier nicht möglich, da grundsätzlich zwei Objekte an einer Beziehung beteiligt sind. Für den Simulator ist es dadurch auch möglich die Beziehungen unabhängig von den Objekten zu verarbeiten.

Wie auch bei den Objekten tritt für Beziehungen ebenfalls die Problematik auf, dass eine Vielzahl unterschiedlicher Arten existiert. Unterschieden werden kann hier zwischen logischen und physikalischen Beziehungen. Während für logische Beziehungen, wie z. B. eine Master-Slave-Beziehung häufig weniger Informationen übertragen werden müssen, werden für physikalische Verbindungen wie z. B. Datenverbindungen abhängig von der Simulationsart und dem Detaillierungsgrad deutlich mehr Informationen bzw. Parameter benötigt. Eine der grundlegendsten Informationen ist dabei die Richtung in die die Verbindung besteht. Diese Angabe entscheidet darüber, wie die Daten über das Netzwerk gesendet werden dürfen. So kann die Datenübertragung beispielsweise nur von Komponente A zu B erfolgen. In der Kommunikationstechnik wird dabei zwischen drei verschiedenen Arten unterschieden – Simplex, Halbduplex, Duplex.

Im Simplex-Betrieb können Daten lediglich in eine Richtung gesendet werden, also beispielsweise nur von Objekt A zu Objekt B. Der Halbduplex-Betrieb ermöglicht es dagegen, die Kommunikation in beide Richtungen durchzuführen. Hierbei kann allerdings immer nur ein Objekt Daten übertragen, sodass die Objekte immer nur abwechselnd Daten senden können. Anders ist dies im sogenannten Duplex-Betrieb. Dieser ermöglicht es, den verbundenen Objekten gleichzeitig Daten über dieselbe Verbindung zu senden. Die Wartezeiten, während ein anderes Objekt Daten überträgt, entfallen daher vollständig (Zenk 2004).

Neben der Datenübertragung müssen aber auch Informationen wie z. B. die Übertragungsgeschwindigkeit oder die Kabellänge angegeben werden können. Wie bereits bei den Objekten sind die relevanten Parameter abhängig von der jeweiligen Simulationsart und dem Detaillierungsgrad des Modells. Um die Flexibilität der Schnittstelle zu gewährleisten müssen zu einer Verbindung daher auch beliebige Parameter angegeben werden können. Hierzu wird das gleiche Prinzip wie bei den Objekten verwendet.

Um festzulegen, zwischen welchen Komponenten die Verbindung hergestellt wird, werden im oben dargestellten XML-Element die Namen und die IDs der jeweiligen Objekte eingetragen. Dadurch ist eine eindeutige Identifizierung der an der Verbindung beteiligten Komponenten durch den Simulator möglich. Die Unterscheidung der Objekte durch *from* und *to* bezieht sich an dieser Stelle nicht auf die Übertragungsrichtung der Daten, sie wird lediglich für das Unterelement *direction* benötigt. Hier kann zwischen l*eft*, *right* und *both* ausgewählt werden, wodurch die Übertragungsrichtung der Verbindung festgelegt wird. Im Fall von *left* erfolgt die Datenübertragung von links nach rechts, also von der Komponente, die

im Element *from* steht, zu der Komponente im Element *to*. Im umgekehrten Fall *right* erfolgt die Kommunikation dagegen in die entgegengesetzte Richtung, also von rechts nach links. Ohne die Unterscheidung von *from* und *to* wäre hier eine genaue Festlegung der Richtung der Datenübertragung nicht möglich, sodass, bei einem Import der Daten in einen Simulator die Verbindung falsch interpretiert werden könnte.

## 4 Durchführung einer Simulationsstudie

Für die Durchführung einer Simulationsstudie wurde die definierte Schnittstelle beispielhaft in den Simulator OMNeT++ implementiert und ermöglicht es so auf die Daten der CMDB i-doit von Synetics zuzugreifen. Mit der Umsetzung ist es möglich Dateien mit den Daten einer CMDB einzulesen und diese in ein entsprechendes Modell des Simulators umzuwandeln. Da die Entwicklung der Schnittstelle nicht an eine spezielle Aufgabe oder Experiment gebunden ist, soll hier beispielhaft ein fiktives und einfaches Stromnetz verwendet werden. In Abbildung 2 ist der Aufbau dieses Netzes in OMNeT++ dargestellt. Vorab ist es allerdings notwendig das Modell in i-doit bzw. einer XML-Datei als Datenaustauschformat abzubilden, um den Import der Daten überhaupt durchführen zu können.

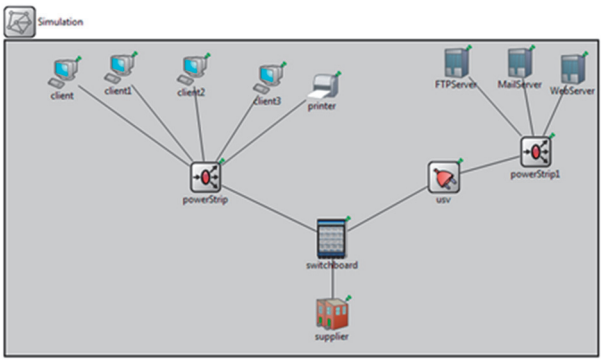


***Abbildung 2:*** *Simulationsmodell*

In diesem Simulation Szenario soll der Ausfall des Stromerzeugers simuliert werden, wodurch alle Komponenten, die nicht an die USV angeschlossen sind ausfallen. Das erwartete Ergebnis dieses Szenarios ist, dass die Simulation den Ausfall erkennt und alle betroffenen Komponenten, die keinen Strom mehr erhalten, markiert und ausfallen lässt. Darüber hinaus muss die USV die Versorgung der angeschlossenen Komponenten übernehmen, wodurch der Ausfall an diesen nicht bemerkbar wird. Daraus resultierend fallen an der USV weiterhin die Stromverbräuche der angeschlossenen Komponenten an, was in diesem Fall durch den Ausfall dem Gesamtverbrauch in der Simulation entspricht.

Für die Durchführung der Simulationsstudie ist es notwendig, dass die Schnittstelle die erforderliche Konfigurationsdatei, siehe Algorithmus 2, erzeugt.

***Algorithmus 2:*** *Konfigurationsdatei der Simulation*

```
[General]
network=Simulation

**.consumption = 100
**.powerStrip.max=1000
**.powerStrip1.max=1000
**.switchboard.max=5000

**.supplier.enabled=false
```

Der Verbrauch der Komponenten ist hier allgemein auf 100 Watt festgelegt und die maximale Leistung der Infrastruktur-Komponenten ist so gewählt, dass sie in dieser Simulation nicht erreicht wird. Weiterführend ist der Stromerzeuger entsprechend der Simulationsaufgabe deaktiviert.

Basierend auf der oben getroffenen Konfiguration erzeugt der Simulator das in Abbildung 3 dargestellte Ergebnis der Simulation.

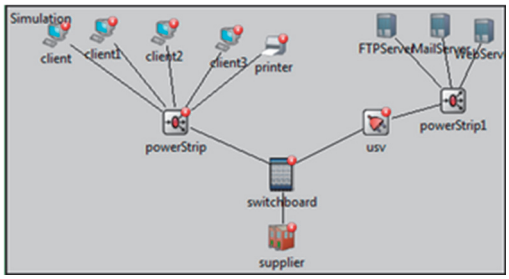


***Abbildung 3****:Simulationsergebnis*

Wie zu erkennen ist, sind in der Simulation alle Komponenten, die nicht an die USV angeschlossen sind ausgefallen, was durch das Symbol in der rechten oberen Ecke der Komponenten dargestellt ist. Da die USV vom Ausfall des Stromerzeugers auch betroffen ist, ist diese ebenfalls durch das Symbol gekennzeichnet. Alle folgenden Komponenten sind dagegen vom Ausfall nicht betroffen, da die USV diese weiterhin mit Strom versorgt. Dies ist zudem in der Simulation ersichtlich, da der Stromverbrauch an der USV 300 Watt beträgt, was dem Stromverbrauch der drei angeschlossenen Server entspricht. Der Stromverbrauch, der während der Simulation beim eigentlichen Stromerzeuger gemessen wurde beträgt dagegen 0 Watt.

Die Ergebnisse, die die Simulation dieses Szenarios ergeben hat, entsprechen den oben genannten erwarteten Ergebnissen, die die Simulation im Idealfall gemäß dem realen System erzielen muss. Neben dem Faktor Gesamtverbrauch des Stromnetzes ist dies auch durch die dargestellten Ausfälle in der Simulation ersichtlich. Daraus resultierend stimmt das Verhalten des Modells mit dem des eigentlichen Systems überein, weshalb die Simulation für dieses Szenario verlässliche Ergebnisse erzeugt.

Bei der Betrachtung der Schritte einer Simulationsstudie nach Law und Kelton (1997). Wird außerdem verständlich, dass der Aufwand für den Anwender deutlich reduziert wurde:

1. Problem Formulierung und Definition des Systems/Modells
2. Wahl von Metriken, Faktoren und Stufen
3. Sammeln von Daten und Modellierung
4. Wahl der Simulationsumgebung, Implementierung des Modells und Verifikation
5. Validierung und Empfindlichkeitsanalyse
6. Durchführung der Experimente, Analyse und Präsentation (Law und Kelton 1997)

Die Schritte zwei bis fünf entfallen vollständig und Schritt eins beschränkt sich nur noch auf die Problem Formulierung, da das System bereits durch die CMDB vorgegeben ist. Der Aufwand für die Durchführung einer Simulationsstudie ist somit durch die Schnittstelle deutlich verkürzt worden.

## 5 Fazit

Mit der hier definierten Schnittstelle ist es möglich die Daten einer CMDB für einen Simulator nutzbar zu machen. Damit dies allgemein für alle CMDBs möglich ist, muss sich diese allerdings erst als Standardschnittstelle etablieren, was ebenfalls auf der Seite der Simulatoren erfolgen muss. Wie die Implementierung in den Simulator OMNeT++ gezeigt hat, müssen die Daten, welche über die Schnittstelle übergeben wurden, in ein Simulator-spezifischen Modell umgewandelt werden. Im Fall von OMNeT++ bedeutet dies neben dem eigentlichen Modell eine entsprechende Konfigurationsdatei, in der die Parameter der Objekte angegeben werden, zu erstellen. Da die Modelle nicht standardisiert sind, muss die Schnittstelle in jeden Simulator einzeln integriert werden, um diese nutzbar zu machen.

Weitergehend müssen für die Schnittstelle Objekt- und Beziehungsarten festgelegt werden, damit die eindeutige Bestimmung von Objekt- und Beziehungsarten für alle Simulatoren und CMDBs gleich ist und keine Differenzen auftreten können. Dafür ist es erforderlich einen Katalog mit allen Arten zu entwerfen und an Änderungen anzupassen.

In weiteren Arbeiten muss die Umsetzung der hier definierten Schnittstelle als Webservice betrachtet werden. Dadurch wäre ein direkter Zugriff auf die Daten der CMDB möglich. Die vorab Umwandlung und das Exportieren der Daten als Datei würde vollständig entfallen und es müssten nur noch für die jeweilige Simulation relevante Modelldaten aus der CMDB ausgelesen werden.

## Literatur

Buchsein, R.: IT-Management mit ITIL® V3. Strategien, Kennzahlen, Umsetzung. Wiesbaden: Vieweg 2007.

ISO/IEC 20000-1: Information technology – Service Management – Part 1: Specification. 2005

Kresse, M.; Bause, M.: ITIL v3 – Alles was man wissen muss. Bad Homburg: Serview 2011.

Law, A.; Averill, M.: Simulation modeling and analysis. New York: McGraw-Hill 2006.

Ray, E. T.: Einführung in XML. Köln: O'Reilly 2004.

Wischki, C.: ITIL V2, ITIL V3 und ISO/IEC 20000. Gegenüberstellung und Praxisleitfaden für die Einführung oder den Umstieg. München, Wien: Hanser 2009.

Zenk, A.: Lokale Netze. Planung, Aufbau, Wartung: München, Boston: Addison-Wesley 2004.