Institut für Architektur von Anwendungssystemen
Universität Stuttgart
Universitätsstraße 38
D-70569 Stuttgart

Diplomarbeit Nr. 3681

Ein Modellierungswerkzeug für BPMN4TOSCA

Thomas Michelbach

Studiengang:	Softwaretechnik
Prüfer:	Prof. Dr. Frank Leymann
Betreuer:	Dipl.-Inf. Oliver Kopp Dipl.-Inf. Uwe Breitenbücher
Begonnen am:	10.07.2014
Beendet am:	09.01.2015
CR-Klassifikation:	H.4.1, H.5.2

Kurzfassung

Das Paradigma des Cloud Computings führt bei der Bereitstellung von IT-Systemen zu einer großen Menge an heterogenen und verteilten Komponenten, die kombiniert werden müssen, um die gesamtheitliche Funktionalität einer Anwendung zu realisieren. Neben der Planung solch komplexer Anwendungstopologien stellt insbesondere das Management der Anwendung einen großen Kosten- und Zeitfaktor dar. Die OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA) beschreibt einen Standard für die Bereitstellung und das Management von Cloud Anwendungen. Dieser beinhalten über die Anwendungstopologie hinaus Managementpläne, die sämtliche Abläufe rund um deren Provisionierung, Wartung und Deprovisionierung in Form von Workflows beschreiben.

Im Rahmen dieser Arbeit wird ein browserbasiertes Modellierungswerkzeug konzipiert und implementiert um Managementpläne für TOSCA-Anwendungen zu erstellen. Die Modellierung erfolgt hierbei auf Grundlage der BPMN-Erweiterung BPMN4TOSCA. Diese ermöglicht durch die Erweiterung der Standard BPMN-Palette, um vier TOSCA-spezifische Elemente, eine stärkere Anbindung zwischen Managementplänen und Anwendungstopologien. Darüber hinaus wird das Modellierungswerkzeug in das bereits existierende Open Source OpenTOSCA-Ökosystem integriert. Insbesondere steht hierbei eine Optimierung des Modellierungswerkzeuges für die Nutzung auf unterschiedlichen Endgeräten im Mittelpunkt.

Inhaltsverzeichnis

Abbildungsverzeichnis

1. Einleitung

1.1 Motivation und Aufgabenbeschreibung

Um moderne IT-Systeme zu realisieren wird vermehrt das Paradigma des Cloud-Computings [1] eingesetzt, dessen Hauptaugenmerk auf der bedarfsgerechten Nutzung und kosteneffizienten Bereitstellung durch Drittanbieter liegt. Die Infrastruktur wird somit nicht länger in eigenen Rechenzentren betrieben sondern, unter Umständen, zu einem bis mehreren Anbietern ausgelagert. Hierbei werden folgende fünf Schlüsseleigenschaften als charakteristisch für das Cloud-Computing eingestuft: On-demand Self Service, Broad Network Access, Resource Pooling, Rapid Elasticity sowie Measured Services [1].

Während sich dieser Wechsel in erster Instanz als kostengünstig erweist, hat sich jedoch der heterogene Charakter verschiedener Komponenten als neue Herausforderung herauskristallisiert [2, 3]. Darüber hinaus hat sich die enge Bindung an den gewählten Cloud-Computing-Anbieter und dessen bereitgestellte Technologie als unerwünschte Nebenerscheinung erwiesen. Diese unerwünschte Bindung und die damit verbundene Abhängigkeit von einem einzelnen Cloud-Computing-Anbieter werden als Vendor-Lock-In bezeichnet. Ein wichtiges Anliegen betrifft somit sowohl die Portabilität von Anwendungen selbst als auch von Prozessen rund um deren Bereitstellung und Wartung [4, 5].

Die Automatisierung dieser Managementaufgaben zählt zu den Grundvoraussetzungen um die oben genannten Schlüsseleigenschaften moderner Cloud-Anwendungen erfüllen zu können. Mit der Topology and Orchestration Specification for Cloud Applications (TOSCA) [6] existiert ein Standard zur Definition von Cloud-Anwendungen sowie den zugehörigen Managementaufgaben. Innerhalb einer TOSCA-Anwendung werden unterschiedliche Managementaufgaben in Form von Managementplänen realisiert. Diese basieren in TOSCA auf standardisierten Workflow-Sprachen und lassen sich somit in grafischer Form modellieren und in einer entsprechenden Umgebung automatisch verarbeiten. Mit BPMN4TOSCA [7] existiert eine Erweiterung der generischen Workflow-Sprache BPMN [8] um TOSCA-spezifische Elemente, die eine stärkere Anbindung zwischen Anwendungstopologien und Managementplänen ermöglicht. Innerhalb des OpenTOSCA-Ökosystems, einer Open Source Implementierung von TOSCA, steht mit der Winery eine zentrale browserbasierte Verwaltung zur Verfügung. Jedoch existiert bis dato in der Winery kein spezielles Modellierungswerkzeug für eine grafische Modellierung von Managementplänen auf Grundlage der BPMN4TOSCA-Erweiterung.

Ziel der vorliegenden Arbeit ist es daher ein solches grafisches Werkzeug zur Modellierung von Managementplänen zu konzipieren und zu implementieren. Dabei soll auf das bestehende OpenTOSCA-Ökosystem, aufgebaut und dieses entsprechend um ein grafisches Werkzeug erweitert werden. Die Modellierung soll hierbei auf Grundlage der BPMN-Erweiterung BPMN4TOSCA erfolgen.

1.2 Aufbau und Kapitelübersicht

Im Folgenden wird die Struktur der vorliegenden Arbeit erläutert. Die Struktur leitete sich dabei aus der Vorgehensweise bei der Entwicklung des Werkzeuges ab und ist wie folgt gegliedert:

Kapitel 1 – Einleitung: Kapitel 1 beschreibt die Aufgabenstellung dieser Arbeit und zeigt die Relevanz der Automatisierbarkeit von Managementaufgaben für Cloud-Anwendungen auf.

Kapitel 2 – Grundlagen: Kapitel 2 erläutert Grundlagen aus dem Umfeld dieser Arbeit. Hierzu zählen neben Begrifflichkeiten aus dem TOSCA-Umfeld auch Grundlagen aus dem Bereich der Workflow-Sprachen.

Kapitel 3 – Methode zur Modellierung von Managementplänen auf Grundlage von BPMN4TOSCA: Kapitel 3 beschreibt die Vorgehensweise bei der Erstellung von TOSCA-Managementplänen durch eine Methode.

Kapitel 4 – Umsetzung der Methode im Rahmen des OpenTOSCA-Ökosystems: Kapitel 4 beschreibt das konkrete Vorgehen bei der Erstellung von TOSCA-Managementplänen speziell für das OpenTOSCA-Ökosystem.

Kapitel 5 – Vereinfachung der Modellierung von Managementplänen mit BPMN4TOSCA: Kapitel 5 schildert die Weiterentwicklung der zuvor eingeführten BPMN-Erweiterung BPMN4TOSCA.

Kapitel 6 – Modellierung von BPMN4TOSCA 2.0 Modellen: Kapitel 6 beschreibt die Entwicklung eines Modellierungswerkzeuges auf Grundlage der in Kapitel 5 vorgestellten Erweiterung von BPMN4TOSCA.

Kapitel 7 – Architektur und technische Realisierung: Kapitel 7 beschreibt die Architektur des in dieser Arbeit entwickelten Modellierungswerkzeuges.

Kapitel 8 – Zusammenfassung und Ausblick: Kapitel 8 fasst die Ergebnisse dieser Arbeit zusammen und bietet einen Ausblick auf mögliche Erweiterungen auf Grundlage dieser Ergebnisse.

2. Grundlagen

In diesem Kapitel werden einige grundlegende Konzepte sowie Technologien erläutert deren Verständnis die Grundlage für den weiteren Verlauf dieser Arbeit bildet. Hierzu wird in Abschnitt 2.1 der Beschreibungsstandard TOSCA für Cloud-Anwendungen eingeführt. Anschließend werden mit BPMN und BPEL [9] zugehörige Workflow-Sprachen vorgestellt. Darauf folgt mit OpenTOSCA [10] (Abschnitt 2.4) die Vorstellung einer Open Source Implementierung von TOSCA. Im Anschluss wird in einer Diskussion (Abschnitt 2.5) der Status Quo der Erstellung von Managementplänen mit den bis dato gegebenen Werkzeugen thematisiert, bevor abschließend die domänenspezifische Sprache BPMN4TOSCA eingeführt wird (Abschnitt 2.6).

2.1 TOSCA

Bei TOSCA [6] – der Topology and Orchestration Specification for Cloud Applications – handelt es sich um einen von OASIS[1] verabschiedeten Standard für die Beschreibung von Cloud-Anwendungen. TOSCA adressiert insbesondere die zuvor genannten Problemfelder des automatisierten Managements von Anwendungen, der Portabilität sowie Interoperabilität von Anwendungen sowie die Wiederverwendbarkeit von Anwendungskomponenten. TOSCA definiert hierzu mit Anwendungstopologien und Managementplänen zwei grundlegende Konzepte (siehe Abbildung 1), die in TOSCA mit Hilfe der Auszeichnungssprache XML formalisiert werden.

Anwendungstopologien beschreiben die strukturellen Aspekte einer TOSCA-Anwendung durch zugehörige Komponenten und deren Beziehungen zueinander. Die Beschreibung erfolgt hierbei auf Grundlage wiederverwendbarer Komponenten. Mit Hilfe von Managementplänen, welche auf standardisierten Workflow-Sprachen basieren, lassen sich sämtliche Managementaufgaben dieser Anwendungstopologien beschreiben. Durch den Einsatz standardisierter Workflow-Sprachen kann die automatische Ausführung dieser Managementpläne in verschiedenen TOSCA-Laufzeitumgebungen gewährleistet werden. Die Definition von Managementaufgaben in Form von Managementplänen sowie deren automatische Ausführbarkeit sorgen somit für eine hohe Wiederverwendbarkeit und eine niedrige Fehlerquote bei der Ausführung. Für die Verwaltung einer TOSCA-Anwendung ist somit, durch den Einsatz von automatisch ausführbaren Managementplänen, kein IT-Spezialwissen mehr erforderlich [4, 5].

[1] http://www.oasis-open.org

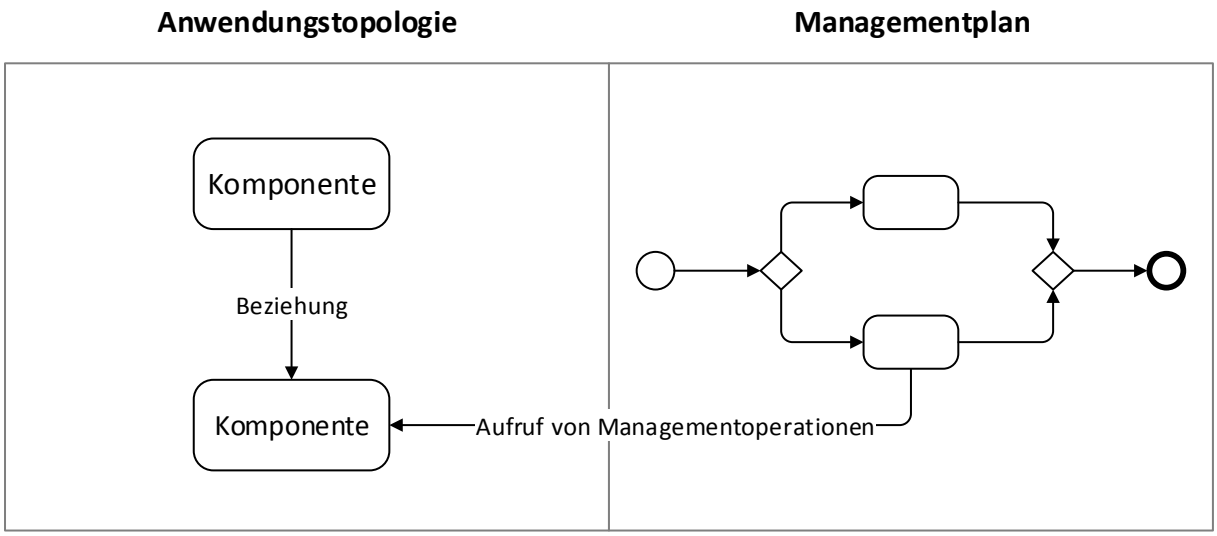

Abbildung 1: Zusammenhang der TOSCA Konzepte (nach [4])

Darüber hinaus definiert TOSCA ein Datenaustauschformat in dem sowohl Anwendungstopologien und Managementpläne als auch sämtliche notwendigen Artefakte einer Anwendung paketiert und distribuiert werden können. Die Ausführung von TOSCA-Anwendungen erfolgt anschließend innerhalb einer TOSCA-Laufzeitumgebung, welche die zur Provisionierung und Verwaltung notwendigen Funktionen bietet.

2.1.1 Topologien

Moderne Cloud-Anwendungen setzen sich häufig aus mehreren eigenständigen Komponenten zusammen. Im Kontext einer TOSCA-Anwendung werden diese Komponenten sowie deren Beziehungen und Abhängigkeiten zueinander durch sogenannte (Anwendungs-)Topologien in Form eines gerichteten Graphen beschrieben. Dieser Graph repräsentiert zu diesem Zeitpunkt jedoch keine konkreten Instanzen der TOSCA-Anwendung sondern beschreibt lediglich, durch typisierte Knoten und Kanten, das strukturelle Umfeld einer instanziierbaren TOSCA-Anwendung. Während ein Knoten dieser Topologie einer Komponente der TOSCA-Anwendung entspricht, beschreibt eine Kante zwischen zwei Knoten die Beziehung bzw. Abhängigkeit der Komponenten zueinander. Innerhalb der Topologie einer TOSCA-Anwendung werden diese Knoten als Node- und die Kanten als Relationship Templates bezeichnet, welche durch ihre jeweilige Typisierung die Zugehörigkeit zu einer speziellen Klasse symbolisieren.

Der Node Type eines Node Template spezifiziert somit nicht nur die Zugehörigkeit zu dieser Klasse sondern definiert auch dessen Eigenschaften (Properties) und zur Verfügung stehende öffentliche Methoden (Operations). Gleichermaßen beschreibt der Relationship Type eines Relationship Templates die Semantik einer Beziehung sowie deren mögliche Eigenschaften (Properties). Eine gerichtete Kante in der Topologie entspricht folglich einem Relationship Template und definiert durch seinen Relationship Type und die Richtung der Kante die Beziehung zwischen zwei Node Templates. Die erläuterten Zusammenhänge zwischen den unterschiedlichen TOSCA-Komponenten werden in Abbildung 2 in grafischer Form dargestellt.

Mit Hilfe von Node- bzw. Relationship Types lassen sich somit die zu Grunde liegenden Strukturen von Node- bzw. Relationship Types einer TOSCA-Anwendung definieren. Jedoch beschreiben diese noch keine konkrete Implementierung. Hierzu sind weitere Artefakte einer TOSCA-Anwendung notwendig. Mit Hilfe von Implementation Artifacts (IA) erfolgt die konkrete Implementierung eines Node Types. Hierzu zählt insbesondere die Implementierung der definierten Operationen eines Node Types. Darüber hinaus werden durch Deployment Artifacts (DA) die für eine Instanziierung notwendigen Artefakte eines Node Types definiert. Im folgenden Beispiel können diese u.a. das eingesetzte Ubuntu-Image für den Ubuntu Node Type als auch die Binaries des MySQL Database Node Types sein.

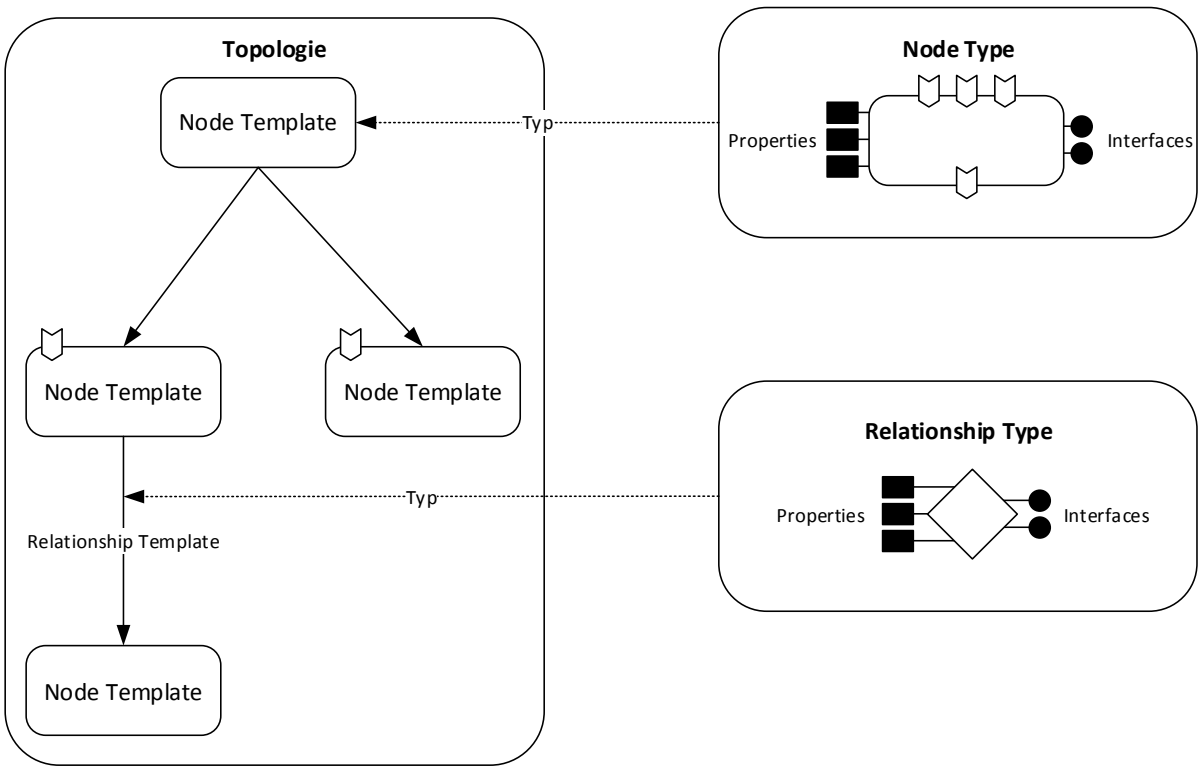

Abbildung 2: Zusammenhang zwischen Topologie-Komponenten (nach [6])

Zum strukturellen Umfeld einer exemplarischen LAMP-Anwendung (Linux-Apache-PHP-MySQL) gehören neben der Anwendung selbst (PHP Application), eine virtuelle Maschine (VM) für ein Linux-Betriebssystem (UbuntuOS), eine MySQL-Datenbank (MySqlDatabase) sowie ein Apache-Webserver (ApacheWebserver). Eine denkbare Topologie einer LAMP TOSCA-Anwendung besteht somit aus entsprechenden Node Types für die fünf zuvor erwähnten Komponenten sowie aus passenden Relationship Types um die Beziehungen zwischen den Komponenten zu beschreiben. In diesem Fall sind für Relationship Types „verbindet-sich-mit“ (*connectsTo*) oder „ist-gehosted-auf“ (*hostedOn*) denkbar. Die Properties und Operations der verschiedenen Node Templates werden durch deren Node Types definiert. Beispielsweise stellt ein Node Template vom Node Type ApacheWebserver Methoden zum Starten (*start*) und Stoppen (*stop*) des Webservers bereit, während ein Node Type vom Typ UbuntuOS Methoden zum Aktualisieren des zugehörigen Paketverwaltungssystems (*updateAPT*) verfügt. Gleichermaßen ist es möglich dass ein VM Node Type Eigenschaften wie die IP-Adresse (*ipAddress*) bereitstellt während sowohl ApacheWebserver als auch MySqlDatabase die entsprechende Angabe zur Port-Nummer (*port*) zur Verfügung stellen.

Node- bzw. Relationship Types einer TOSCA-Anwendung sind nicht zwangsweise an eine konkrete Topologie gebunden, sondern können in weiteren Topologien erneut zum Einsatz kommen. Somit kann ein hoher Grad an Wiederverwendbarkeit gewährleistet werden. Die Möglichkeit sowohl Node- als auch Relationship Types hierarchisch vererben zu können, führt zudem zu verschiedenen denkbaren Graden der Abstraktion bei der Erstellung von Node- bzw. Relationship Types. Um auf Grundlage einer vorliegenden Topologie eine konkrete TOSCA-Anwendung provisionieren zu können, müssen die hierfür notwendigen Managementaufgaben definiert werden. Die Topologie selbst beschreibt hierbei nicht ausschließlich das strukturelle Umfeld einer TOSCA-Anwendung, sondern definiert explizit über die durch Node Types bereitgestellten Operations die Managementfähigkeiten einer Topologie. Die konkreten Managementaufgaben werden hingegen mit Hilfe von Managementplänen formuliert.

2.1.2 Managementpläne

Die zuvor beschriebenen TOSCA-Topologien beschreiben sämtliche Komponenten einer TOSCA-Anwendung und die Beziehung dieser Komponenten zueinander. Die Verwaltung dieser Topologien wird durch Managementaufgaben beschrieben. Unter den Begriff einer Managementaufgabe fallen in diesem Zusammenhang sämtliche organisatorischen Aufgaben, die die Verwaltung einer TOSCA-Anwendung, basierend auf der zuvor modellierten Topologie, beschreiben. Die hierfür notwendigen Managementpläne beschreiben folglich Managementaspekte einer TOSCA-Anwendung die sich aus der Abfolge einzelner auszuführender Managementaufgaben zusammensetzen. Die Abfolge der Aufgaben kann hierbei sequentiell und/oder parallel erfolgen. Somit stellen Managementplänen Abläufe dar, die sich grafisch mit Hilfe von gerichteten Graphen und damit in Form von Flussdiagrammen beschreiben lassen.

Folglich handelt es sich um Workflows [11], die mit Hilfe einer Workflow-Sprache beschrieben werden können. Bei der Nutzung einer Workflowsprache zur Formulierung eines Managementplans kann daher von den Möglichkeiten sowie Eigenschaften dieser Gebrauch gemacht werden. Zu diesen Eigenschaften zählen u.a. die parallele Ausführung von Aktivitäten sowie das detaillierte Monitoring eines Workflows. Darüber hinaus können robuste und etablierte Workflow-Engines genutzt werden um diese Managementpläne zu verarbeiten.

Das Spektrum an Managementaspekten einer TOSCA-Anwendung reicht von der Provisionierung, über die Verwaltung bis hin zu deren Deprovisionierung [12]. Managementpläne sind somit Workflows, die bestimmte Phasen aus dem Lebenszyklus einer TOSCA-Anwendung beschreiben. Zu diesem Zweck wird zwischen drei Klassen von Managementplänen (im weiteren Sinne) unterschieden: Build-, Management- sowie Termination-Pläne [6]. Während Build-Pläne Workflows zur Provisionierung einer TOSCA-Anwendung beschreiben, werden in Termination-Plänen Workflows zur Deprovisionierung von TOSCA-Anwendungen beschrieben. Mit Hilfe von Management-Plänen im engeren Sinne lassen sich Workflows rund um das Management (d.h. die Verwaltung) einer bereits bereitgestellten TOSCA-Anwendung beschreiben.

Vereinfacht gesagt repräsentiert jeder Knoten im Graph eines Managementplans eine Managementaufgabe während eine gerichtete Kante zwischen zwei Knoten den zeitlichen und logischen Ablauf zwischen diesen Aufgaben definiert. Ferner existieren spezielle Start- und Endknoten, die den Beginn bzw. das Ende eines Managementplans definieren. Mit Hilfe von Verzweigungen die über mehrere ausgehende Kanten verfügen, lässt sich der parallele Ablauf mehrerer simultaner Aufgaben beschreiben. Bei der Zusammenführung der Verzweigungen wird entsprechend auf die Beendigung aller eingehenden Kanten gewartet bevor der ausgehende Ablauf fortgeführt wird. Abbildung 3 zeigt den Ausschnitt eines vereinfachten Managementplans welcher sich aus sechs einzelnen Managementaufgaben auf Grundlage des geschilderten Konzeptes zusammensetzt.

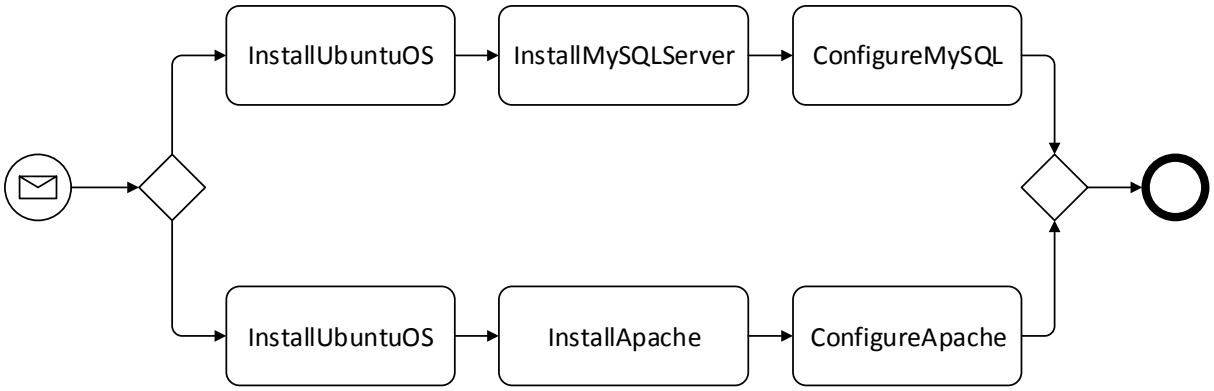

Abbildung 3: Ausschnitt eines vereinfachten Managementplans in BPMN

Die TOSCA-Spezifikation [6] schlägt mit BPEL (Abschnitt 2.3) und BPMN (Abschnitt 2.2) explizit die Nutzung zweier weit verbreiteter Standards aus dem Bereich der Workflow-Sprachen zur Beschreibung von Managementplänen vor. Die Nutzung einer standardisierten Workflow-Sprache gewährleistet die Portabilität und die Interoperabilität zwischen verschiedenen Laufzeitumgebungen.

2.1.3 CSAR

Ein Service Template fasst die beiden zuvor beschriebenen strukturellen Aspekte einer TOSCA-Anwendung – in Form einer Topologie (Abschnitt 2.1.1) – sowie deren Managementaspekte – in Form von Managementplänen (Abschnitt 2.1.2) – zusammen und bildet somit die Grundlage einer TOSCA-Anwendung.

TOSCA-Anwendungen werden in Cloud Service Archiven (CSAR) bereitgestellt. Hierbei handelt es sich um ein standardisiertes Archiv-Format, welches sämtliche notwendige Artefakte und Ressourcen einer TOSCA-Anwendung beinhaltet. Hierzu zählen neben einem Service Template die zugehörigen Node- bzw. Relationship Types, deren Definition entweder im selben Service Template erfolgt oder welche aus einem separaten Dokument in dieses importiert wird. Darüber hinaus enthält das CSAR einer TOSCA-Anwendung alle notwendigen Deployment- (DA) sowie Implementation-Artifacts (IA), die für die Bereitstellung einer TOSCA-Anwendung notwendig sind.

Während die TOSCA-Spezifikation das Format und die Struktur des CSAR definiert, werden keine Aussagen über die Beschaffung der einzelnen Artefakte, insbesondere nicht von Topologien und Managementplänen getätigt. Die Erstellung von Managementplänen und die eingesetzten Werkzeuge sind somit nicht Teil der TOSCA-Spezifikation. Folglich ist es dem Modellierer freigestellt verschiedene Werkzeuge zur Erstellung einer TOSCA-Anwendung zu nutzen.

Mit Hilfe eines einzelnen CSAR (und der darin enthaltenen Daten) lassen sich somit komplette TOSCA-Anwendungen und die zugehörigen Managementaspekte beschreiben. Diese Paketierung ermöglicht es u.a. komplette TOSCA-Anwendungen über einen entsprechenden Marktplatz zu distribuieren [13], welche anschließend lediglich durch eine TOSCA-Laufzeitumgebung provisioniert werden müssen. Für die Bereitstellung eines vorhandenen CSAR ist eine TOSCA-Laufzeitumgebung notwendig. Diese ist für die Ausführung und Verwaltung der im CSAR enthaltenen Artefakte verantwortlich.

2.1.4 TOSCA Laufzeitumgebung

TOSCA-Laufzeitumgebungen sind in der Lage TOSCA-Anwendungen in Form eines CSAR zu verarbeiten. Die TOSCA-Laufzeitumgebung stellt hierbei nicht nur die im CSAR enthaltene Anwendung bereit sondern bietet darüber hinaus eine standardisierte API, welche zur Anbindung von IAs genutzt werden kann. Bei der Verarbeitung von TOSCA-Anwendungen kann die TOSCA-Laufzeitumgebung neben der Berücksichtigung von funktionalen Anforderungen ebenfalls für die Einhaltung nicht-funktionaler Anforderungen verantwortlich sein. Mit Policy4TOSCA wurde sowohl ein Vorgehen für die Formalisierung von nicht-funktionalen Anforderungen in Form von Policies vorgestellt, als auch ein konkreter Ansatz der Implementierung demonstriert [14].

Bei der Verarbeitung von CSAR sind mit imperativer sowie deklarativer Verarbeitung zwei unterschiedliche Arten der Verarbeitung durch eine TOSCA-Laufzeitumgebung möglich [6]. Imperative Verarbeitung basiert auf dem Einsatz von Managementplänen (Abschnitt 2.1.2) die sowohl die notwendigen Aufgaben selbst als auch deren Reihenfolge klar definieren. Der Ansatz der deklarativen Verarbeitung hingegen verlagert die Verantwortung der Abbildung von Managementaspekten auf die Laufzeitumgebung. Folglich ist diese verantwortlich für die Ausführung der Managementaufgaben und muss bei diesem Ansatz, durch Interpretation der vorliegenden Topologie, die notwendigen Schritt selbst ableiten um diese automatisch in der korrekten Reihenfolge ausführen zu können. Der Unterschied zwischen beiden Ansätzen liegt somit im Grad der Flexibilität und dem Aufwand der Modellierung. Wird ein imperativer Ansatz verfolgt, müssen neben der Topologie zur Provisionierung, Verwaltung und Deprovisionierung entsprechende Managementpläne modelliert werden. Hierdurch kann zwar ein hoher Grad an Flexibilität gewährleistet werden, jedoch ist der Aufwand im Vergleich zum deklarativen Ansatz weitaus höher. In letzterem Ansatz reicht die reine Modellierung einer Topologie aus. Ein Mittelweg zwischen diesen beiden Möglichkeiten bilden hybride Ansätze.

Im Rahmen des OpenTOSCA-Ökosystems (Abschnitt 2.4) wurde mit dem Plan Generator eine Komponente implementiert, die durch Kombination von deklarativem sowie imperativem Vorgehen einen solchen Ansatz verfolgt. Hierbei wird automatisch ein ausführbarer Build-Plan aus einer vorliegenden Topologie generiert, welcher anschließend durch weitere manuelle Anpassungen individualisiert werden kann [15]. Die Wahl des richtigen Ansatzes ist abhängig von der gewählten TOSCA-Laufzeitumgebung und der Komplexität der Anwendung.

2.2 BPMN 2.0

Die Business Process Modeling and Notation Version 2.0 (nachfolgend kurz: BPMN) ist eine grafische Notation zur visuellen Beschreibung von Workflows. Der BPMN-Standard definiert sämtliche Elemente sowie das zugehörige Regelwerk an Kombinationsmöglichkeiten, die für die grafische Modellierung von Workflows notwendig sind. BPMN verfolgt bei der Modellierung eines Workflows einen Graph-orientierten Ansatz, d.h. ein Workflow wird mit Hilfe von Knoten, Kanten sowie den zugehörigen Transitionen zwischen den Knoten beschrieben.

Der Sequenzfluss zeigt hierbei die Reihenfolge an in der die einzelnen Aktivitäten des Workflows abgearbeitet werden. Hierzu sind diese Aktivitäten über gerichtete Sequenzflusskanten verbunden. Neben dem Sequenzfluss wird der Datenfluss als zentraler Bestandteil bei der Modellierung des Workflows betrachtet. Dieser Datenfluss beschreibt lesende oder schreibende Zugriffe von Aktivitäten oder Ereignissen auf Daten. In BPMN muss dieser Datenfluss explizit modelliert werden. Hierfür stehen spezielle Datenobjekte zur Verfügung, die über Datenassoziationen mit Aktivitäten und Ereignissen verbunden werden können.

Insbesondere die Eigenschaft der Erweiterbarkeit ermöglicht die Nutzung und Anpassung von BPMN für Workflows jeder Art. Diese Eigenschaft und die leicht verständliche, aber dennoch mächtige grafische Notation machen BPMN zum de facto Standard für die grafische Modellierung von Workflows. Die einfach verständliche und wenig technische Notation von BPMN ermöglicht zudem eine Verständigung zwischen verschiedenen Beteiligten an einem Workflow.

2.3 WS-BPEL 2.0

Web Services Business Process Execution Language Version 2.0 (nachfolgend kurz: WS-BPEL 2.0) [9] ist eine Beschreibungssprache für ausführbare Workflows. Im Gegensatz zu BPMN beherrscht BPEL neben einer graphbasierten Modellierung ebenfalls eine Block-strukturierte Modellierung [16], d.h. die Kontrollstruktur eines Workflows wird durch die Verschachtelung der Aktivitäten ausgedrückt. Im Falle von BPEL werden die einzelnen Aktivitäten eines WS-BPEL 2.0 Prozesses durch Webservices implementiert. In BPEL modellierte Workflows sind somit explizit für eine nachfolgende Ausführung konzipiert, während BPMN, dank seiner einfachen visuellen Verständlichkeit, häufig auch zu Dokumentationszwecken genutzt wird.

BPEL selbst definiert im Gegensatz zu BPMN keine eigene grafische Notation, sodass bei der grafischen Modellierung von BPEL die Darstellung des Workflows stark vom eingesetzten Werkzeug abhängt und folglich nicht standardisiert ist. Daher wird für die grafische Modellierung häufig BPMN eingesetzt, welches anschließend zur Ausführung in BPEL umgewandelt wird.

An dieser Stelle sei angemerkt, dass BPMN seit der Version 2.0 ebenfalls die Möglichkeit bietet in BPMN modellierte Prozesse auszuführen. Jedoch ist die Transformation oder die direkte Ausführung von BPMN 2.0 lediglich in eingeschränkter Form möglich, sodass BPEL sich im Unternehmensbereich (für die Ausführung modellierter Prozesse) weiterhin einer großen Beliebtheit erfreut [17].

2.4 OpenTOSCA

OpenTOSCA [10] wurde maßgeblich am Institut für Architektur von Anwendungssystemen der Universität Stuttgart entwickelt. Bei OpenTOSCA handelt es sich um ein Open Source Ökosystem für TOSCA-Anwendungen, welches aus mehreren eigenständigen Komponenten besteht. Das OpenTOSCA-Ökosystem (siehe Abbildung 4) besteht aus den Komponenten Winery [18], dem OpenTOSCA-Container [10] sowie der der Vinothek [19]. Winery bezeichnet ein grafisches Werkzeug zur Verwaltung von TOSCA-Anwendungen. Beim OpenTOSCA-Container handelt es sich um eine Open Source Laufzeitumgebung für TOSCA-Anwendungen. Mit der Vinothek steht im OpenTOSCA-Ökosystem ein Self-Service Portal für TOSCA-Anwendungen bereit.

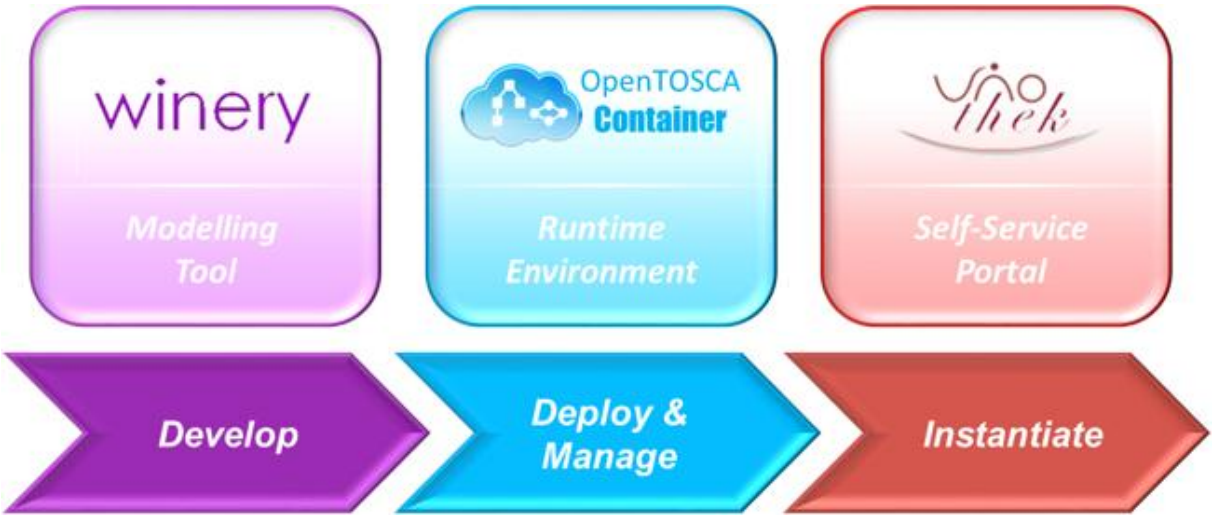

Abbildung 4: Übersicht des OpenTOSCA-Ökosystems (aus [20])

2.4.1 OpenTOSCA-Container

Der OpenTOSCA-Container ist eine Open Source Implementierung einer TOSCA-Laufzeitumgebung und verfolgt bei der Verarbeitung von TOSCA-Anwendungen einen imperativen Ansatz und basiert somit auf dem Einsatz von Managementplänen zur Abbildung von Managementaufgaben. Folglich müssen für jegliche Managementaspekte der TOSCA-Anwendung, die notwendigen Managementpläne im CSAR mitgeliefert werden.

Diese Managementpläne werden mit Hilfe von Plug-ins verarbeitet, welche in der Lage sind unterschiedliche Workflow-Sprachen zu verarbeiten. Zu diesem Zeitpunkt existiert jedoch lediglich ein Plug-in für die automatische Verarbeitung von Managementplänen im BPEL-Format.

2.4.2 Vinothek

Self-Service Portale ermöglichen Benutzern den Zugriff auf bestimmte, üblicherweise komplexe, technische Funktionsabläufe auch ohne tiefgreifendes technisches Verständnis. Bei der Vinothek handelt es sich um ein browserbasiertes Self-Service Portal, welches Benutzern die Möglichkeit bietet TOSCA-Anwendungen über ein einheitliches Webinterface in verschiedenen TOSCA-Laufzeitumgebungen zu provisionieren [19]. Dem Benutzer bleiben hierbei spezielle Eigenarten und technische Details der jeweiligen TOSCA-Laufzeitumgebung verborgen, sodass mit Hilfe der Vinothek verschiedene TOSCA-Laufzeitumgebungen eingebunden und verwaltet werden können ohne über Kenntnisse in Bezug auf technische Aspekte verfügen zu müssen.

Die Anbindung an die jeweilige TOSCA-Laufzeitumgebung erfolgt hierbei über spezielle Plug-ins, welche die Anbindung an diese entsprechend abstrahieren. Im Rahmen der OpenTOSCA-Laufzeitumgebung erfolgt die Provisionierung einer TOSCA-Anwendung auf Grundlage von Managementplänen (Abschnitt 2.1.2), sodass das entsprechende Plug-in zur Provisionierung einer TOSCA-Anwendung den Prozess der Ausführung dieser Pläne in der OpenTOSCA Workflow-Engine regelt.

2.4.3 Winery

Bei Winery handelt es sich ebenfalls um ein browserbasiertes Werkzeug aus dem OpenTOSCA-Ökosystem zur Erstellung und Verwaltung von TOSCA-Anwendungen. Winery bietet eine grafische Benutzerschnittstelle (GUI) zur Modellierung von Topologien (Topology Modeler GUI) per Drag-and-Drop (siehe Abbildung 5). Darüber hinaus stellt die Winery Werkzeuge zur Verwaltung von in der Topologie enthaltenen Komponenten wie Node- oder Relationship Types (Element Management GUI) sowie einen integrierten XML-Editor zur Be- und Überarbeitung von XML-Dateien bereit.

Sämtliche Informationen werden in einem Repository gespeichert, das über eine REST-Schnittstelle zum Import und Export von Daten in und aus externen Anwendungen verfügt. Zur Bereitstellung modellierter Anwendungen für eine TOSCA Laufzeitumgebung unterstützt Winery durch das Repository einen Export im CSAR-Format.

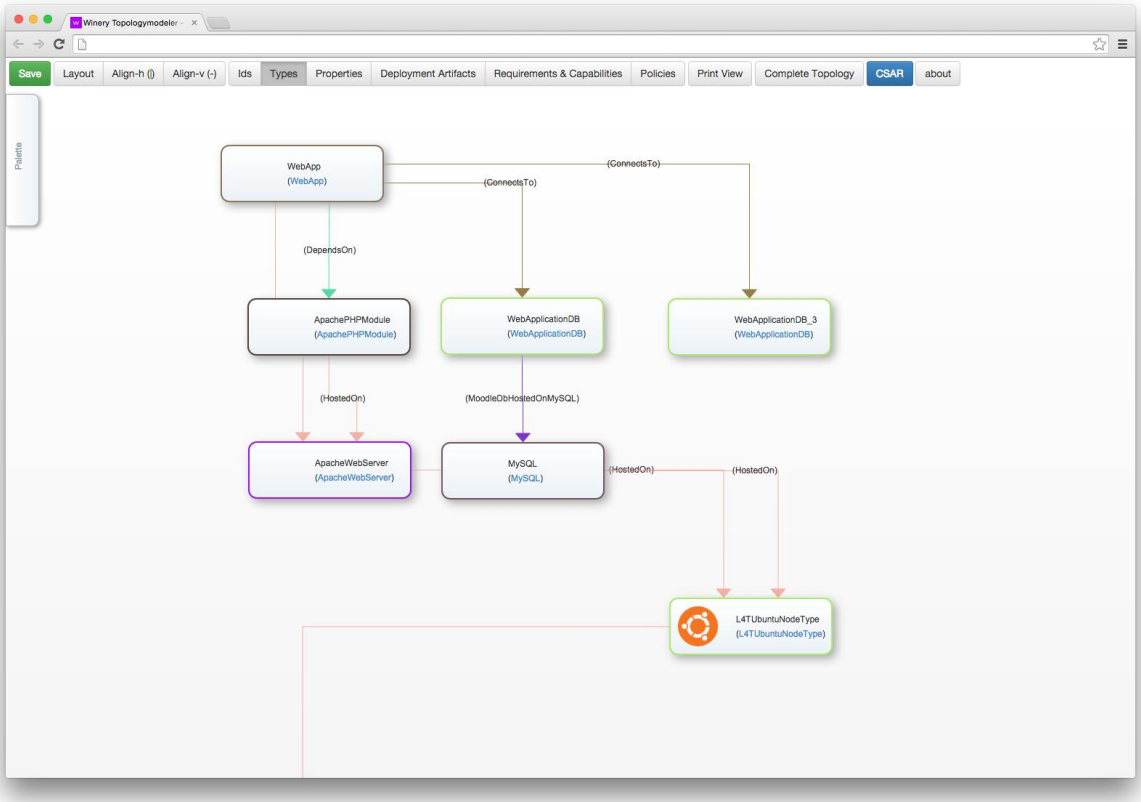

Abbildung 5: Topology Modeler GUI von Winery

2.5 Diskussion: Erstellung von Managementplänen

Der Ansatz der imperativen Verarbeitung von CSAR durch die OpenTOSCA-Laufzeitumgebung führt zu einer erheblichen Relevanz von Managementplänen für eine TOSCA-Anwendung. Liegen diese Managementpläne in wohldefinierter Form vor kann durch eine TOSCA-Laufzeitumgebung auf dieser Grundlage automatisch eine TOSCA-Anwendung provisioniert werden, da diese einer klaren Ausführungssemantik folgen.

Die Modellierung dieser Pläne stellt jedoch eine erhebliche Herausforderung dar. Managementpläne basieren in OpenTOSCA auf der komplexen Beschreibungssprache XML. Der Managementplan muss damit entweder händisch formuliert oder mit Hilfe eines entsprechenden Werkzeuges grafisch modelliert und anschließend als XML gespeichert werden. Im Falle der manuellen Formulierung kommen häufig proprietäre textbasierte Werkzeuge zum Einsatz, die den Modellierer bei der Modellierung eines Managementplans wenig unterstützen können. Fortgeschrittene Editoren, die eine automatische Vervollständigung oder das Syntax-Highlighting unterstützen, erleichtern durch diese Merkmale die Modellierung und können somit zu einer zeitlichen Ersparnis beitragen.

Hierdurch wird der Modellierer zwar bei der Vermeidung von syntaktischen Fehlern unterstützt, die strukturelle Korrektheit von Managementplänen ist jedoch weiterhin allein von den Kenntnissen und Fähigkeiten des Modellierers abhängig. In diesem Aspekt bieten die fortgeschritten Editoren demzufolge keinerlei Vorteile gegenüber den proprietären Werkzeugen.

Die händische Formulierung von Managementplänen im XML-Format bleibt somit zeitintensiv und fehlerbehaftet. Während für die grafische Modellierung von Topologien mit Winery ein entsprechendes Werkzeug bereitgestellt wird, existieren zu diesem Zeitpunkt keine spezifischen Werkzeuge für die grafische Modellierung von TOSCA-Managementplänen in Winery. Bei BPMN handelt es sich um eine grafische Notation zur Beschreibung von generischen Workflows, eine besondere Eignung speziell für die grafische Darstellung von TOSCA-Managementplänen ist jedoch ohne eine entsprechende Erweiterung der Sprache nicht vorhanden. Folglich lässt sich grundsätzlich, insbesondere mit Hilfe von grafischen Werkzeugen für BPMN, ein Managementplan für TOSCA-Anwendungen erstellen. Jedoch stellen entsprechende grafische Werkzeuge ausschließlich für generisches BPMN relevante Elemente zur Verfügung, sodass bei der Modellierung eine zusätzliche Abstraktionsebene speziell für die Modellierung von Managementplänen fehlt. Die in TOSCA-Managementplänen formulierten Workflows beschreiben häufig lesende und schreibende Zugriffe einzelner Properties von Nodes und Relationships durch Aktivitäten. Lesende bzw. schreibende Zugriffe auf diese Eigenschaften lassen sich mit Hilfe von Aufrufen auf die OpenTOSCA Container API realisieren. Diese müssen jedoch für jeden Zugriff explizit durch den Einsatz von BPMN Service Tasks modelliert werden.

Es lässt sich somit festhalten, dass die grafische Modellierung mit Hilfe von BPMN-kompatiblen Werkzeugen im Gegensatz zur händischen Formulierung in XML zwar eine geringere Fehlerquote aufweist (da der XML-Code durch das Werkzeug generiert wird), jedoch weiterhin vertiefende Kenntnisse und Fähigkeiten des Modellierers erfordert. Die grafische Modellierung mit Hilfe einer generischen Workflow-Sprache bleibt ohne domänenspezifische Elemente somit sehr aufwendig.

2.6 BPMN4TOSCA

Bei BPMN4TOSCA [7] handelt es sich um den Vorschlag einer domänenspezifischen Auszeichnungssprache zur Modellierung von Managementplänen für TOSCA-Anwendungen. BPMN4TOSCA adressiert primär die im Abschnitt 2.5 beschriebenen Probleme, die bei der Modellierung eines Managementplans mit einer generischen Workflow-Sprache wie BPMN auftreten. Im Mittelpunkt steht somit eine direktere Integration des Managementplans an die modellierte TOSCA-Topologie sowie an deren zugehörige Eigenschaften und Operationen.

BPMN4TOSCA macht von der Erweiterbarkeit von BPMN Gebrauch und führt insgesamt vier neue BPMN-Elemente ein: den TOSCA Topology Management Task (Abschnitt 2.6.1), den TOSCA Node Management Task (Abschnitt 2.6.2), den TOSCA Script Task (Abschnitt 2.6.3) sowie das TOSCA Data Object (Abschnitt 2.6.4).

Jedes dieser vier Elemente erweitert ein bereits bestehendes BPMN-Element und lässt sich über definierte Abläufe wieder in standardkonformes BPMN zur Nutzung in einer TOSCA-Laufzeitumgebung umwandeln. Folglich müssen keine Veränderungen an der TOSCA-Laufzeitumgebung selbst vorgenommen werden. Lediglich das entsprechende Modellierungswerkzeug muss eine Unterstützung für BPMN4TOSCA bieten. Im Folgenden werden die vier neuen Elemente von BPMN4TOSCA vorgestellt sowie deren Vorteile bei der Modellierung eines Managementplans erläutert.

2.6.1 TOSCA Topology Management Task

Abbildung 6: TOSCA Topology Management Task Icon (aus [7])

Mit Hilfe des TOSCA Topology Management Tasks (siehe Abbildung 6) ist es möglich Topology Management Operationen, die durch den TOSCA-Container bereitgestellt werden, direkt aufzurufen. Hierzu erweitert der TOSCA Topology Management Task den BPMN Service Task, der dazu genutzt wird die entsprechenden Aufrufe an die Schnittstelle des TOSCA-Containers zu senden.

2.6.2 TOSCA Node Management Task

Abbildung 7: TOSCA Node Management Task Icon (aus [7])

Der TOSCA Node Management Task (siehe Abbildung 7) erweitert ebenfalls den BPMN Service Task um eine direktere Anbindung an Operationen. Jedoch handelt es sich in diesem Fall nicht um Operationen die durch den TOSCA-Container, sondern um Operationen die von Nodes in der Topologie bereitgestellt werden. Die Definition dieser Operationen erfolgt, zusammen mit den jeweiligen Ein- und Ausgabeparametern der Operation, in den zugehörigen Node Types. Die konkrete Implementierung erfolgt mit Hilfe von IA's welche die tatsächliche Funktionalität der Operation bereitstellen.

2.6.3 TOSCA Script Task

Abbildung 8: TOSCA Script Task Icon (aus [7])

Mit Hilfe des TOSCA Script Tasks (siehe Abbildung 8) ist es möglich Skripte auf Knoten der Topologie auszuführen. Das Skript selbst kann hierbei als Teil des Tasks definiert werden oder auf ein vorhandenes Skript auf einem Topologie Knoten verweisen. Um mit Hilfe eines TOSCA Script Tasks automatisch ein Skript auf einem Knoten ausführen zu können definiert BPMN4TOSCA ein Interface mit den Operationen *deployScript*, *runScript* sowie *undeployScript*, das der jeweilige Node bereitstellen muss.

BPMN4TOSCA definiert hierbei lediglich die Interfaces sowie die zugehörigen Parameter, nicht die konkrete Handhabung der Skripte. Um ein Skript auf einen Node zu kopieren wird dieses als Eingabeparameter an die *deployScript* Operation eines Nodes übergeben und liefert als Ausgabeparameter eine eindeutige Id zurück. Diese Id wird als Eingabeparameter an die *runScript* Operation übergeben, welche das Skript ausführt und als Ausgabeparameter die Rückgabe des Skripts enthält. Mit Hilfe der *undeployScript* Operation, die als Eingabeparameter wiederum die Id der *deployScript* Operation erhält, ist es möglich das Skript vom entsprechenden Knoten zu entfernen. Der TOSCA-Container regelt hierbei die korrekte zeitliche Ausführung der zuvor erwähnten Operationen mit den zugehörigen Parametern auf dem entsprechenden Node.

2.6.4 TOSCA Data Object

Abbildung 9: TOSCA Data Object Icon (aus [7])

Das TOSCA Data Object (siehe Abbildung 9) erweitert die Funktionalität des BPMN Data Objects um einen direkten Lese- und Schreibzugriff auf Node Properties sowie Relationship Properties der Topologie. Diese Zugriffe auf die Node- bzw. Relationship Properties können mit Hilfe einer Schnittstelle des TOSCA-Containers realisiert werden. Die hierfür notwendigen Aufrufe erfolgen im Falle von lesenden bzw. schreibenden Zugriffen über BPMN Service Tasks, welche die notwendigen Informationen vom TOSCA-Container lesen bzw. an diesen senden. Bei der Nutzung eines TOSCA Data Objects entfällt die Modellierung dieser zusätzlichen BPMN Service Tasks, sodass lediglich ein einziges TOSCA Data Object platziert werden muss, um auf die Properties eines Nodes bzw. einer Relationship zugreifen zu können. Hierbei ist es durchaus denkbar mehrere TOSCA Data Objects zu platzieren, die Properties derselben Node- bzw. Relationship referenzieren. Das TOSCA Data Object erweitert das Standard BPMN Data Object um die Eigenschaften *nodeTemplateId* bzw. *relationshipTemplateId*, welches die entsprechenden Verweise auf das zu referenzierende Node Template bzw. Relationship Template setzt. Über die optionalen Eigenschaften *nodeInstanceId* bzw. *relationshipInstanceId* können konkrete Instanzen adressiert werden falls die modellierte Toplogie mehrere Instanzen desselben Templates erlaubt.

Ein initialer Prototyp eines Modellierungswerkzeuges auf Grundlage von BPMN4TOSCA wurde von Kopp et al. [7] vorgestellt. Dieser basierte auf einer Vorgängerversion von Winery namens Valesca[2]. In dieser Arbeit wird ein Modellierungswerkzeug vorgestellt, dass (i) BPMN4TOSCA erweitert, (ii) in Winery integriert ist sowie (iii) auf mobilen Plattformen betrieben werden kann.

[2] http://www.cloudcycle.org/valesca/

2.7 Ansätze der Transformation zwischen BPMN 2.0 und WS-BPEL 2.0

In den vorherigen Kapiteln wurde beschrieben dass es sich bei BPMN und BPEL um zwei häufig verwendete Workflow-Sprachen handelt, die zu unterschiedlichen Zwecken eingesetzt werden können. Eine Transformation von BPEL nach BPMN verfolgt demnach das Ziel das Modell eines ausführbaren Workflows in ein einfach zu bearbeitendes und verständliches grafisches Modell umzuwandeln. Gleichermaßen wird umgekehrt versucht mit Hilfe einer Transformation ein grafisches Modell in einen ausführbaren Workflow umzuwandeln. Da beiden Workflow-Sprachen fundamental andere Modelle zu Grunde liegen und die beiden Sprachen nicht über die gleiche Ausdrucksmächtigkeit verfügen, existieren für diese Transformation unterschiedliche Ansätze. Mendeling et al. [21] beschreiben vier Vorgehensweisen für eine Transformation zwischen BPMN und BPEL, die im Folgenden vorgestellt werden.

Bei der *Element-Preservation*, dem einfachsten Ansatz, wird jedes Element des Prozessgraphen, und damit insbesondere auch die Kanten zwischen den Elementen, auf ein entsprechendes WS-BPEL 2.0 Block-Konstrukt abgebildet. Die Kanten zwischen den Elementen wiederum werden auf *link*-Elemente abgebildet. Es handelt sich somit um eine 1:1 Abbildung zwischen den Modellen, die jedoch auf Grund der Einfachheit der Abbildung mehr Elemente als notwendig enthält und somit vergleichsweise schwer verständlich ist. Die Grundvoraussetzung für diesen Ansatz ist das Vorliegen eines azyklischen BPMN-Graphen.

Die *Element-Minimization* baut auf der vorangehend vorgestellten Vorgehensweise auf. Leere Aktivitäten werden jedoch aus der Abbildung entfernt. Diese resultieren aus der Transformation von Gateways, die in diesem Ansatz durch Links bzw. Join-Conditions ersetzt werden. Da die *Element-Minimization* auf *Element-Preservation* basiert bildet ein azyklischer BPMN 2.0 Graph ebenfalls die Grundvoraussetzungen für diesen Ansatz.

Beim Ansatz der *Structure-Identification* wird versucht die aus einem BPMN-Prozessgraph identifizierten Muster auf ähnlich strukturierte BPEL-Aktivitäten abzubilden. Grundvoraussetzung hierfür ist das Vorliegen eines strukturierten Graphen für die Transformation.

Structure-Maximization nutzt den vorangehenden Ansatz der *Structure-Identification* rekursiv um die maximale Struktur im Prozessgraphen extrahieren zu können. Die verbleibende Struktur wird entsprechend des *Element-Preservation* Ansatzes transformiert. Somit müssen in diesem Ansatz folglich insgesamt zwei Strategien implementiert werden.

2.8 Verwandte Arbeiten

Da es sich bei BPMN4TOSCA um eine neuartige domänenspezifische Sprache handelt, existieren neben dem zuvor erwähnten Prototypen (siehe Abschnitt 2.6) keine weiteren speziellen Werkzeuge welche explizit eine Modellierung in BPMN4TOSCA ermöglichen. Jedoch finden sich verwandte Werkzeuge welche durchaus für eine grafische Modellierung von Managementplänen für TOSCA respektive OpenTOSCA in Frage kommen.

Bei Oryx[3] handelt es sich um ein browserbasiertes Modellierungswerkzeug, welches am Hasso-Plattner-Institut[4] (HPI) entwickelt wurde. Insbesondere durch den Einsatz von sog. Stencil-Sets, einer Menge von modellierbaren Elementen und deren Beziehungen, erlaubt Oryx die Erweiterung um nahezu beliebige Prozess-Modellierungssprachen. Somit ist prinzipiell eine BPMN4TOSCA-kompatible Modellierung durch die Entwicklung eines eigenen BPMN4TOSCA Stencil-Sets möglich. Mit dem Signavio Process Editor[5] ist jedoch auf Grundlage von Oryx ein kommerzielles Produkt entstanden, sodass Oryx selbst am HPI nicht weiter aktiv betreut wird. Darüber hinaus wurde der bestehende Source Code lediglich unter der GNU GPL v3 Lizenz veröffentlicht, sodass dieser im Sinne einer End-to-End Open Source Toolchain nicht in das OpenTOSCA-Ökosystem eingebunden werden kann.

Beim BPEL-Designer[6] handelt es sich ebenfalls um ein Modellierungswerkzeug speziell für die Modellierung in der Workflow-Sprache BPEL. Durch die direkte Modellierung in BPEL entfällt eine Transformation in eine ausführbare Sprache, sodass theoretisch die Nutzung des BPEL-Designers für die Modellierung von TOSCA-Managementplänen denkbar ist. Zudem ist die Palette des BPEL-Designers um domänenspezifsche Aktivitäten erweiterbar. Beim BPEL-Designer handelt es sich jedoch um ein Eclipse-basiertes Werkzeug, welches nicht ohne weiteres in das browserbasierte Umfeld des OpenTOSCA-Ökosystems integriert werden kann.

Somit lässt sich festhalten dass grundsätzlich die Möglichkeit besteht Managementpläne für TOSCA bzw. OpenTOSCA in händischer Form zu formulieren, die gegebene Problematik hierbei wurde jedoch in Abschnitt 2.5 thematisiert. Die Notwendigkeit Managementpläne in grafischer Form modellieren zu können besteht weiterhin, da zu diesem Zeitpunkt kein entsprechendes grafisches Werkzeug existiert das die notwendige Eignung aufweist. Es lässt sich folglich festhalten dass insbesondere kein Werkzeug existiert welches von den Vorzügen einer domänenspezifschen Sprache wie BPMN4TOSCA Gebrauch macht.

[3] http://bpt.hpi.uni-potsdam.de/Oryx/
[4] http://www.hpi.de/
5 http://www.signavio.com/
[6] http://www.eclipse.org/bpel/

3. Methode zur Modellierung von Managementplänen auf Grundlage von BPMN4TOSCA

Nachdem in den vorangehenden Kapiteln die Grundlagen einer TOSCA-Anwendung erläutert worden sind, wird in diesem Kapitel auf das allgemeine Vorgehen bei der Erstellung eines Managementplans für eine TOSCA-Anwendung eingegangen. Im Mittelpunkt dieser Betrachtung steht explizit die Erstellung eines neuen Managementplans für eine noch nicht existente Topologie. Hierzu werden sämtliche notwendigen Aktivitäten, die letztendlich zur Modellierung eines Managementplans notwendig sind, losgelöst von einer konkreten Technologie in chronologischer Reihenfolge in Form einer Methode betrachtet. Die nachfolgende Abbildung zeigt die einzelnen notwendigen Schritte der Methode. Dabei handelt es sich bei Schritt 4 um eine optionale Maßnahme. Die einzelnen Schritte werden in den folgenden Unterkapiteln erläutert. Die Methode ist dabei vollkommen technologieunabhängig, d.h. sie kann für jede TOSCA-Umgebung angewendet werden. In Kapitel 4 wird anschließend die Methodenanwendung im Rahmen des OpenTOSCA-Ökosystems gezeigt.

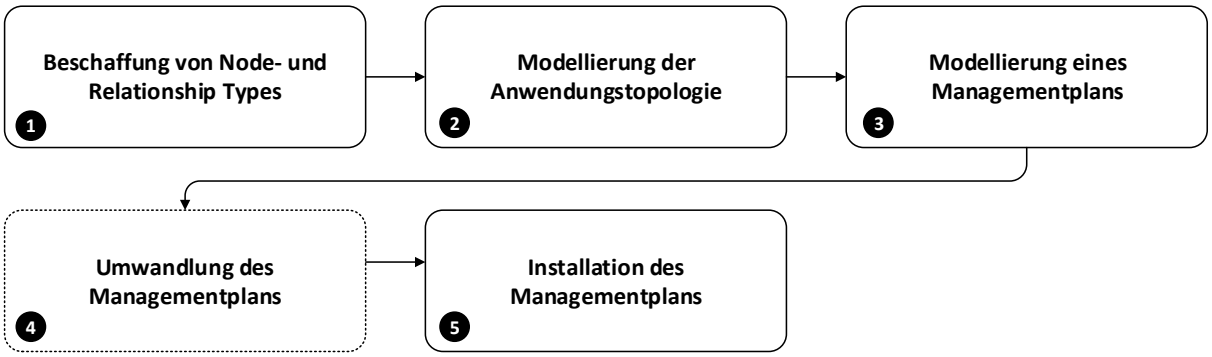

Abbildung 10: Vorgehensmodell zur Erstellung von Managementplänen

3.1 Schritt 1: Beschaffung von Node- und Relationship Types

Beschaffung der benötigten Typdefinitionen zur semantischen Definition der Komponenten und Relationen in der Anwendungstopologie.

Die Grundlage der Modellierung einer Topologie für eine TOSCA-Anwendung bildet insbesondere die Definition von Node- bzw. Relationship Types, welche zur Spezifikation der Semantik der Topologie (Node Templates) sowie deren Relationen (Relationship Templates) verwendet werden. Eine wichtige Rolle spielen in diesem Zusammenhang die Definition von Operations und Properties der Node- bzw. Relationship Types, da diese später im Managementplan aufgerufen und referenziert werden. Ohne eine vollständige Definition von Node- und Relationship Types lässt sich folglich keine provisionierbare Topologie modellieren und somit auch kein zugehöriger Managementplan erstellen.

Die Wiederverwendbarkeit spielt eine große Rolle im TOSCA-Umfeld [4, 5]. Neben der Möglichkeit komplette TOSCA-Anwendungen in Form von CSAR zu distribuieren (Abschnitt 2.1.3) spielt auch die Wiederverwendbarkeit einzelner Komponenten eine zentrale Rolle. Basierend auf der Portabilität des TOSCA-Modells ist es denkbar die notwendigen Node- bzw. Relationship Types aus dem Community-Umfeld des TOSCA-Ökosystems zu beziehen. Darüber hinaus kann der Bezug über die Cloud-Anbieter direkt organisiert werden. Hierzu müssten diese lediglich entsprechende Node- und Relationship Types ihrer eigenen Services für das TOSCA-Ökosystem zur Verfügung stellen [4].

So wurden beispielsweise im Rahmen des CloudCycle-Projekts[7] bereits Node Types für Amazon EC2[8] sowie OpenStack[9] entwickelt, die in neuen TOSCA-Anwendungen wiederverwendet werden können.

Können notwendige Node- bzw. Relationship Types nicht aus dem TOSCA-Ökosystem bezogen werden, ist es dennoch möglich eigene Definitionen wiederzuverwenden, da diese als wohldefinierte wiederverwendbare Bausteine vorliegen. Sollten die notwendigen Node- bzw. Relationship Types nicht zur Verfügung stehen, müssen diese selbst definiert werden. Node- und Relationship Types basieren im Rahmen dieser Arbeit auf der Auszeichnungssprache XML, sodass diese Dateien entweder von Hand formuliert werden müssen oder mit Hilfe eines entsprechenden Werkzeuges erstellt werden können. Die Struktur in der die Modelle anschließend vorliegen müssen lässt sich der TOSCA-Spezifikation [6] entnehmen.

3.2 Schritt 2: Modellierung der Anwendungstopologie

Modellierung der Topologie einer TOSCA-Anwendung auf Grundlage der zuvor beschafften Node- und Relationship Types.

Managementpläne beschreiben Managementaspekte einer konkreten Topologie. Die zuvor beschafften Node- bzw. Relationship Types müssen somit, bevor ein Managementplan modelliert werden kann, in diesem Schritt in Form einer Topologie modelliert werden. Diese Topologie beschreibt die strukturellen Aspekte der TOSCA-Anwendung und die hierfür notwendigen Komponenten basierend auf den zuvor beschafften Node Types sowie deren Beziehungen, entsprechend basierend auf den Relationship Types zueinander.

Da Topologien ebenfalls im XML- oder JSON-Format abgelegt werden kommt für die Erstellung dieser wiederrum eine händische Formulierung oder die Nutzung eines grafischen Werkzeuges zur Modellierung in Frage. Im Gegensatz zur Definition von Node- bzw. Relationship Types bietet sich in diesem Schritt die Nutzung eines grafischen Werkzeuges besonders an, da eine Anwendungstopologie am besten in Form eines Graphen modelliert werden kann. Dieser wird hierbei durch Knoten, welche in TOSCA Komponenten darstellen, und Kanten zur Repräsentation der Relationen modelliert. Die TOSCA-Spezifikation definiert

[7] http://www.cloudcycle.org
[8] http://www.amazon.com/ec2
[9] http://www.openstack.org

hierzu ausschließlich das Schema des Modells in der die Topologie anschließend vorliegen muss, nicht aber das konkret einzusetzende Werkzeug.

3.3 Schritt 3: Modellierung des Managementplans

Formalisierung der Abfolge einzelner Managementaufgaben für die Anwendungstopologie auf Basis einer domänenspezifischen Workflow-Sprache.

In diesem Schritt werden die notwendigen Managementaufgaben einer TOSCA-Anwendung in Form eines Managementplans zusammengetragen. Die zentrale Frage lautet in diesem Schritt, welcher Managementaspekt in diesem konkreten Managementplan abgedeckt werden soll. Die entsprechende Ausrichtung des Managementplans spiegelt sich im Typ des Managementplans (Build-, Management- oder Termination-Plan) wieder. Basierend auf der in Abschnitt 3.2 modellierten Topologie definiert ein Managementplan die Reihenfolge der einzelnen Managementaufgaben, welche zur Ausführung des gewünschten Managementaspekts benötigt werden. Die Aktivitäten selbst entsprechen in der Regel einzelnen Aufrufen von Node Type Operations die in Schritt 3.1 auf den jeweiligen Node Types definiert wurden.

Managementpläne basieren auf Workflow-Sprachen und unterliegen bei einer händischen Formulierung den in Abschnitt 2.5 beschrieben Problemen. Bei einer grafischen Modellierung bietet sich mit BPMN (Abschnitt 2.2) eine Workflow-Sprache mit einer umfangreichen grafischen Notation an. Hierbei handelt es sich jedoch um eine generische Workflow-Sprache, die keine spezielle Unterstützung für TOSCA bietet. Mit BPMN4TOSCA (Abschnitt 2.6) konnte die generische BPMN-Palette um vier neue TOSCA-spezifische Elemente erweitert werden, die eine intuitivere Modellierung von Managementplänen ermöglicht. Für die Modellierung eines Managementplans kommen somit grafische Werkzeuge oder eine händische Formulierung in Frage.

3.4 Schritt 4: Umwandlung des Managementplans

Überführung des Managementplans in ein ausführbares Modell zur automatischen Ausführung der einzelnen Managementaufgaben in einer kompatiblen Laufzeitumgebung.

Managementpläne können grundsätzlich entweder manuell oder automatisch ausgeführt werden. Während bei der manuellen Ausführung jeder Schritt von Hand ausgeführt bzw. angestoßen werden muss, können bei der automatischen Ausführung die einzelnen Aktivitäten automatisch von der TOSCA-Laufzeitumgebung ausgeführt werden. Folglich liegt es nahe eine Möglichkeit zu finden Managementpläne automatisch ausführen zu können. Hierzu müssen die zuvor modellierten Managementpläne entweder in einer ausführbaren Workflow-Sprache vorliegen oder zuvor in eine solche transformiert werden.

Liegt der modellierte Managementplan bereits in einer ausführbaren Workflow-Sprache vor, ist dieser Schritt obsolet und eine Umwandlung in eine ausführbare Workflow-Sprache entfällt. Im Falle einer Modellierung mit BPMN4TOSCA (Abschnitt 3.3) liegen die Managementpläne in keiner ausführbaren Workflow-Sprache vor, da aktuell keine TOSCA-Laufzeitumgebung BPMN4TOSCA als Sprache für Managementpläne unterstützt. Somit müssen diese zuvor in eine ausführbare Workflow-Sprache transformiert werden.

In diesem Schritt ist es denkbar hierfür spezielle Werkzeuge einzusetzen, die in der Lage sind diese Umwandlung automatisch vorzunehmen. Alternativ kann die Umwandlung von Hand durchgeführt werden. In beiden Fällen werden sämtliche Aktivitäten des zuvor modellierten Managementplans auf die entsprechenden Konstrukte einer ausführbaren Workflow-Sprache abgebildet. Ansätze hierfür wurden im Abschnitt 2.7 erläutert.

3.5 Schritt 5: Installation des Managementplans

Verarbeitung des ausführbaren Managementplans zum Management einer TOSCA-Anwendung durch eine kompatible Laufzeitumgebung.

Nachdem sichergestellt wurde, dass der modellierte Managementplan in einer ausführbaren Workflow-Sprache vorliegt, muss die TOSCA-Anwendung in Form eines CSAR paketiert werden. Dieses CSAR enthält nun alle für eine Bereitstellung der Anwendung notwendigen Daten sowie alle Artefakte, welche zur Ausführung der darin enthaltenen Managementpläne benötigt werden. Auf Grundlage dieses CSAR kann die TOSCA-Anwendung von einer TOSCA-Laufzeitumgebung provisioniert und anschließend verwaltet werden.

Die Provisionierung einer TOSCA-Anwendung durch die TOSCA-Laufzeitumgebung erfolgt hierbei basierend auf einem ausführbaren Build-Plan im CSAR. Für alle weiteren Managementaspekte müssen ebenfalls entsprechende Managementpläne im CSAR vorliegen. Zur Ausführung von Managementplänen stellt die TOSCA-Laufzeitumgebung in der Regel eine entsprechende GUI oder eine Schnittstelle zur Verfügung.

4. Umsetzung der Methode im Rahmen des OpenTOSCA-Ökosystems

Nachdem im vorangehenden Kapitel die allgemeine Methode für die Erstellung von Managementplänen diskutiert wurde, wird in diesem Kapitel die spezielle Umsetzung der Methode im Rahmen des OpenTOSCA-Ökosystems konkretisiert. Hierzu werden alle notwendigen Schritte der Methode erneut in chronologischer Reihenfolge (siehe Abbildung 11) betrachtet und dabei insbesondere auf die Eigenschaften des OpenTOSCA-Ökosystems eingegangen.

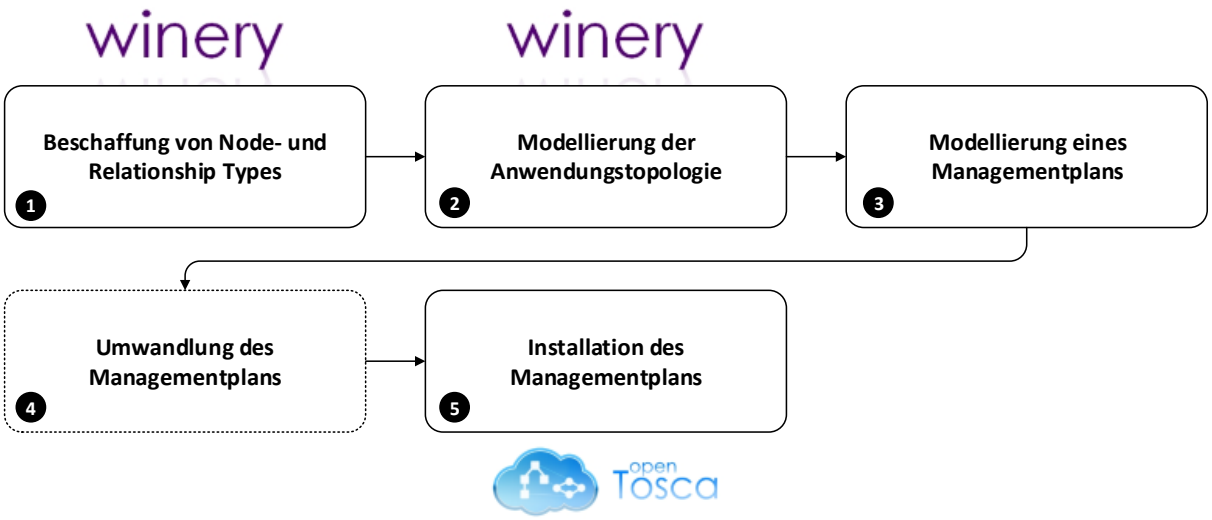

Abbildung 11: Konkretes Vorgehensmodell für OpenTOSCA

4.1 Schritt 1: Beschaffung von Node- und Relationship Types

Mit der Winery (Abschnitt 2.4.3) steht im OpenTOSCA-Ökosystem ein Werkzeug zur Verwaltung und Erstellung von Node- und Relationship Types zur Verfügung. Mit Hilfe der Winery ist es darüber hinaus möglich sämtliche für eine TOSCA-Anwendung notwendigen Aspekte zu verwalten. Die Grundlage der Modellierung einer Topologie bildet hierbei insbesondere die Definition von Node- bzw. Relationship Types. Durch die Portabilität und die durch TOSCA gegebene Abstraktion dieser Komponenten ist es möglich, dass diese entweder aus einer zuvor modellierten TOSCA-Anwendung oder durch die TOSCA-Community beschafft werden können. Liegen für einen Managementplan notwendige Node- bzw Relationship Types nicht vor, können diese in der Winery als neue Node- oder Relationship Types definiert werden. Hierzu werden Node- bzw. Relationship Types durch einem Namen in einem Namensraum identifiziert und anschließend mit Hilfe einer GUI durch entsprechenden Properties ergänzt.

Node Types können zudem mit Hilfe einer weiteren GUI um Operations, die in entsprechenden Interfaces organisiert sind, um deren Input- bzw. Output-Parameter ergänzt werden. Im Kontext der Modellierung mit BPMN4TOSCA muss an dieser Stelle darauf geachtet werden, dass ein entsprechendes Interface, welches die notwendigen Operationen für die Ausführung von Skripten durch den TOSCA Script Task enthält, hinzugefügt wird. Durch die Angabe von Source- und Target-Node Types für Relationship Types kann ausgeschlossen werden, dass nicht kompatible Node Types bei der Modellierung der Anwendungstopologie verbunden werden und somit eine semantisch falsche Verbindung zustande kommt.

Die neuen Node- bzw. Relationship Types werden im Repository der Winery gespeichert. In diesem Repository werden sämtliche Node- und Relationship Types angeboten, welche durch die Winery verwaltet werden. Diese Eigenschaft macht die Winery innerhalb des OpenTOSCA-Ökosystems somit zur zentralen Verwaltung aller zur Verfügung stehender Node- und Relationship Types. Diese stehen sowohl für die Modellierung einer Topologie mit Hilfe der Winery als auch über die REST-Schnittstelle der Winery zur Verfügung.

4.2 Schritt 2: Modellierung der Anwendungstopologie

Die Modellierung einer Topologie für eine TOSCA-Anwendung ist im OpenTOSCA-Ökosystem ebenfalls durch die Winery möglich. Winery verfügt über den Topology-Modeler in welchem diese Modellierung in grafischer Form möglich ist. Die Grundlage dieser Modellierung bilden hierbei die zuvor beschafften Node- und Relationship Types, welche in Form von Node- bzw. Relationship Templates im Topology-Modeler zu einer Topologie modelliert werden. Hierzu muss lediglich ein Service Template in der Service Template Übersicht hinzugefügt und anschließend ausgewählt werden. Im Anschluss lässt sich der Topology Modeler öffnen mit dem die Modellierung einer Topologie in grafischer Form möglich ist. In der Palette des Topology-Modelers werden alle im Service Template verfügbaren Node Types zur Modellierung angeboten. Um ein Node Template zur Topologie hinzuzufügen muss der entsprechende Node Type lediglich per Drag-and-Drop von der Palette auf der Arbeitsfläche positioniert werden. Anschließend können Node Templates durch das Hinzufügen von Requirements, Capabilities und konkreten Werten für Properties annotiert werden. Node Templates können auf der Arbeitsfläche durch Relationship Templates (basierend auf den zuvor beschafften Relationship Types) verbunden werden. Hierbei werden die in den Relationship Types hinterlegten Restriktionen in Bezug auf die Verbindung von Node Types berücksichtigt, sodass eine fehlerhafte Modellierung ausgeschlossen werden kann. Ist die Modellierung abgeschlossen lässt sich die Topologie speichern und falls gewünscht auch direkt in Form eines CSAR paketieren und exportieren.

4.3 Schritt 3: Modellierung eines Managementplans

Derzeit unterstützt Winery kein grafisches Modellierungswerkzeug mit Unterstützung für eine BPMN4TOSCA-kompatible Modellierung. Somit muss manuell ein ausführbarer Managementplan erzeugt werden um diesen auch in der OpenTOSCA-Laufzeitumgebung ausführen zu können. Bei den hierfür notwendigen grafischen Werkzeugen handelt es sich um keine speziellen Werkzeuge für die Erstellung von Managementplänen, sondern um generische Werkzeuge für die Erstellung von BPEL-Workflows. Ebenfalls ist es möglich, Managementpläne von Hand zu formulieren. Auf die Probleme bei dieser Form der Erstellung von Managementplänen wurde in Abschnitt 2.5 eingegangen. Von den Vorzügen der Modellierung basierend auf der BPMN4TOSCA-Erweiterung kann somit zu diesem Zeitpunkt kein Gebrauch gemacht werden. Daher wird im Rahmen dieser Arbeit ein Modellierungswerkzeug auf Grundlage der BPMN4TOSCA-Erweiterung in Winery integriert.

4.4 Schritt 4: Umwandlung des Managementplans

Der OpenTOSCA-Container kann zu diesem Zeitpunkt lediglich Managementpläne im BPEL-Format (Abschnitt 2.3) verarbeiten. Im OpenTOSCA-Ökosystem existiert ferner keine Möglichkeit automatisch BPMN4TOSCA in BPEL für eine Ausführung im OpenTOSCA-Container umzuwandeln. Folglich müssen Managementpläne händisch mit Hilfe eines entsprechenden Werkzeuges direkt in BPEL formuliert oder mit Hilfe von BPMN4TOSCA modelliert und anschließend manuell in BPEL transformiert werden.

Darüber hinaus existieren zu diesem Zeitpunkt keine aktiv betreuten Open Source Projekte die sich mit der Thematik der Transformation zwischen BPMN und BPEL beschäftigen. Es existieren jedoch kommerzielle Lösungen. Mit dem Ziel mit dem OpenTOSCA-Ökosystem selbst eine End-to-End Open Source Toolchain anbieten zu können, werden diese Lösungen im Rahmen dieser Arbeit jedoch nicht betrachtet.

4.5 Schritt 5: Installation des Managementplans

Zuvor modellierte TOSCA-Managementpläne, die in Form von fertigen ausführbaren XML-Dateien vorliegen, können mit Hilfe der Winery zu einem CSAR paketiert und auf diesem Weg für den OpenTOSCA-Container bereitgestellt werden. Hierzu wird der fertige Managementplan mit Hilfe der Winery GUI entsprechend benannt und dem zuvor erstellen Service Template hinzugefügt. Dies geschieht in der Winery GUI innerhalb eines Service Templates. In diesem Schritt kann neben dem Typ (Build-, Management- bzw. Termination-Plan) des Managementplans das vorliegende Format (WS BPEL 2.0 bzw. BPMN 2.0) spezifiziert werden.

An dieser Stelle sei angemerkt, dass der OpenTOSCA-Container zum aktuellen Zeitpunkt zwar die Möglichkeit bietet Managementpläne in verschiedenen Formaten abzulegen, jedoch lediglich in der Lage ist Managementpläne im WS-BPEL 2.0 Format korrekt zu verarbeiten. An dieser Stelle ist es ferner möglich, bestehende Managementpläne zu entfernen oder bestehende Managementpläne durch das Hinzufügen einzelner Input- bzw. Output-Parameter zu bearbeiten. Anschließend kann das fertige CSAR in die OpenTOSCA-Container GUI hochgeladen werden und die Anwendung mit Hilfe dieser provisioniert werden. Darüber hinaus ist es möglich, weitere im CSAR enthaltene Managementpläne auszuführen um auf diesem Weg den weiteren Lebenszyklus der Instanz der TOSCA-Anwendung zu steuern und zu verwalten. Die Managementpläne werden anschließend von der Plan Engine des OpenTOSCA-Containers verarbeitet, der vor einer Ausführung durch die jeweilige Laufzeitumgebung, die Aufrufe im Managementplan an die konkreten Endpunkte bindet, bzw. über den Operation Invoker des OpenTOSCA-Containers realisiert [22, 23].

5. Vereinfachung der Modellierung von Managementplänen mit BPMN4TOSCA

In den vorangehenden Kapiteln wurde sowohl das allgemeine Vorgehen bei der Erstellung eines TOSCA-Managementplans zusammengefasst, als auch das spezielle Vorgehen im Rahmen des OpenTOSCA-Ökosystems. Hierzu wurden alle notwendigen Schritte im Detail betrachtet und falls möglich das detaillierte Vorgehen im entsprechenden Werkzeug geschildert. Als Ergebnis dieser Betrachtung lässt sich festhalten, dass insbesondere für die Schritte 3 und 4 aus Kapitel 4 weder Werkzeugunterstützung noch wohldefinierte Abläufe existieren. Im Mittelpunkt einer Vereinfachung der Modellierung eines Managementplans für OpenTOSCA steht somit die Verwendung der domänenspezifischen Sprache BPMN4TOSCA für die Modellierung eines Managementplans (Abschnitt 4.3). Die damit verbundene Transformation eines nicht ausführbaren Managementplans in ausführbares BPEL (Abschnitt 4.4) für eine automatische Ausführung im OpenTOSCA-Container ist nicht Teil dieser Arbeit. In diesem Kapitel wird daher die Möglichkeit einer Vereinfachung der Modellierung von Managementplänen mit Hilfe von BPMN4TOSCA in Form einer Erweiterung dieser domänenspezifischen Sprache untersucht, bevor im nächsten Kapitel auf das konkrete Modellierungswerkzeug eingegangen wird.

5.1 Anforderungen

Bei der Modellierung mit BPMN muss neben dem Sequenzfluss ebenfalls der Datenfluss explizit modelliert werden. Der Sequenzfluss wird in Form einer durchgezogenen gerichteten Kante zwischen Aktivitäten dargestellt und beschreibt den Ablauf von Aktivitäten. Der Datenfluss hingegen wird mit Hilfe einer gestrichelten gerichteten Kante dargestellt und beschreibt den Informationsfluss zwischen Aktivitäten und Data Objects. Je mehr Abhängigkeiten also zwischen Aktivitäten und Data Objects bestehen, desto höher ist auch der Informationsfluss. Folglich nimmt die Anzahl an gestrichelten Kanten im Prozess zu. Diese Steigerung der Kantenzahl führt zu einer Zunahme der grafischen Komplexität und zu einer Abnahme der Leserlichkeit [24].

Bei der Modellierung eines TOSCA-Managementplans spielt die Modellierung des Informationsflusses eine zentrale Rolle. Beispielweise werden häufig Eigenschaften wie IP-Adressen, Zugangsdaten oder Datenpfade von Node- bzw. Relationship Templates gelesen oder geschrieben. Um die Werte dieser Eigenschaften lesen oder schreiben zu können sind für deren Abruf spezielle BPMN-Service Tasks notwendig. BPMN4TOSCA konnte durch die Einführung des TOSCA Data Objects (Abschnitt 2.6.4) bereits die Zahl hierfür notwendiger Aktivitäten senken, jedoch nicht die Anzahl an Kanten im Prozess. Jedes TOSCA Data Object muss unter Umständen mit mehr als einer Aktivität verbunden werden um den Informationsfluss korrekt darzustellen.

Beispielsweise spielt bei der Instanziierung einer virtuellen Maschine in der Cloud von Amazon Web Services (AWS) neben dem Amazon Maschine Image (AMI) sowohl die Region in der die Instanz gestartet werden soll, als auch der Instanz-Typ, der die Leistungsfähigkeit der Instanz angibt, eine tragende Rolle. Bei der Modellierung mit BPMN4TOSCA wird die Instanziierung selbst in Form eines TOSCA Node Management Tasks modelliert, welcher folglich auf dem AmazonEC2 Node Type die entsprechende Operation zur Instanziierung aufruft. Die oben genannten Daten dienen dieser Aktivität als Eingabeparameter und wurden für diese Operation entsprechend als Eingabeparameter definiert. Ebenso wurde beispielsweise die öffentliche IP-Adresse der neuen Instanz als entsprechender Ausgabeparameter definiert. Um die Eingabeparameter dieses TOSCA Node Management Tasks mit passenden Werten zu befüllen, müssen diese entweder fest hinterlegt werden, oder aus einem zuvor modellierten passenden TOSCA Data Object gelesen werden. Gleichermaßen muss der Ausgabeparameter des TOSCA Node Management Tasks, falls dieser im weiteren Verlauf des Managementplans genutzt werden soll, in ein passendes TOSCA Data Object geschrieben werden. Somit muss dieser beispielhafte TOSCA Node Management Task für jeden Eingabeparameter über eine eingehende und für jeden Ausgabeparameter entsprechend mit einer ausgehenden Kante mit den entsprechenden Eigenschaften eines passenden TOSCA Data Objects verbunden werden. Abbildung 12 zeigt die geschilderte Verknüpfung zwischen TOSCA Node Management Tasks und TOSCA Data Objects, die in diesem Beispiel aus drei eingehenden sowie einer ausgehenden Kante besteht.

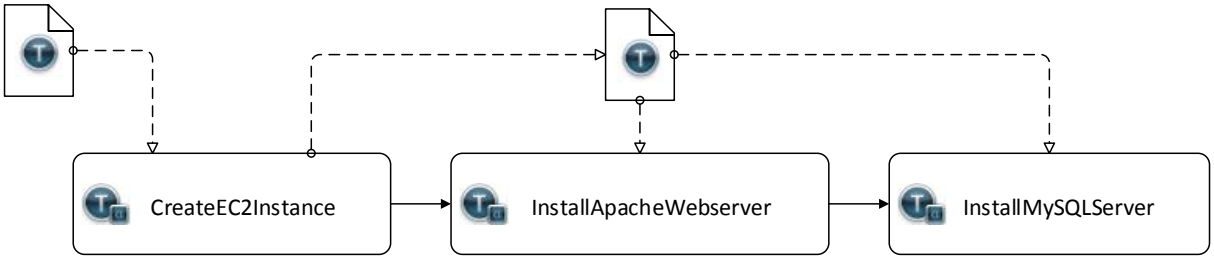

Abbildung 12: Verknüpfung von TOSCA Data Objects mit TOSCA Node Management Tasks

Aus diesem Beispiel lassen sich drei Problemfelder identifizieren: (i) werden (TOSCA) Data Objects eingesetzt so ist, ohne ein tieferes Verständnis für den Managementplan, beim reinen Betrachten der ein- und ausgehenden Kanten des (TOSCA) Data Objects, nicht ersichtlich welche Eigenschaften referenziert werden. (ii) Je mehr Aktivitäten lesenden oder schreibenden Zugriff auf ein (TOSCA) Data Object benötigen, desto schwerer fällt es die optimale Position des (TOSCA) Data Objects zu finden, ohne die Lesbarkeit durch sich kreuzende Kanten zu beeinflussen. (iii) Je mehr (TOSCA) Data Objects eingesetzt werden, desto höher ist die grafische Komplexität des Managementplans.

5.2 BPMN4TOSCA 2.0

Da die Entwicklung von TOSCA respektive OpenTOSCA seit der Entwicklung von BPMN4TOSCA fortgeschritten ist, müssen diese neuen Gegebenheiten bei der Entwicklung eines Modellierungswerkzeuges auf Grundlage von BPMN4TOSCA berücksichtigt werden. Darüber hinaus wurden im vorangehenden Abschnitt einige Problemfelder bei der Modellierung mit BPMN4TOSCA erläutert, die nun in diesem Abschnitt diskutiert werden. Um die grafische Komplexität eines Managementplans zu reduzieren ist es folglich hilfreich, die Anzahl von BPMN-Elementen auf dem Managementplan zu reduzieren. Insbesondere zählt hierzu die Anzahl der (TOSCA) Data Objects inklusive der zugehörigen ein- bzw. ausgehenden Kanten. Auf Grundlage der im Abschnitt 5.1 formulierten Anforderungen an eine Vereinfachung von BPMN4TOSCA wird im Folgenden BPMN4TOSCA 2.0 vorgestellt. Hierzu wird im ersten Schritt BPMN4TOSCA reduziert (Abschnitt 5.2.1) und im zweiten Schritt ein neues Modellierungskonzept (Abschnitt 5.2.2) vorgestellt.

5.2.1 Reduktion von BPMN4TOSCA

Um die Modellierung weiter vereinfachen zu können, müssen die vorhandenen Konzepte betrachtet und ggf. reduziert werden. Aus den von Kopp et al. [7] formulierten Anforderungen an ein BPMN4TOSCA-kompatibles Modellierungswerkzeug wurden in Abschnitt 2.6 vier neue BPMN4TOSCA-Elemente vorgestellt. Im Folgenden werden diese Elemente und die zugehörigen Anforderungen, die zu deren Einführung führen, auf eine weitere Vereinfachung hin untersucht.

Bei einer genaueren Betrachtung der Definition eines TOSCA Script Tasks lässt sich feststellen, dass es sich hierbei lediglich um die Verkettung dreier TOSCA Node Management Tasks handelt. Bei den durch das Interface bereitgestellten Operationen *deployScript*, *runScript* sowie *undeployScript* handelt es sich um Operationen die, genauso wie normale TOSCA Node Management Operationen, durch den Node Type bereitgestellt werden. Dies macht den TOSCA Script Task obsolet, da dieser in Form eines IA implementiert und über den Aufruf eines entsprechenden TOSCA Node Management Tasks ausgeführt werden kann. Folglich kann dieses Element, im Rahmen einer Vereinfachung, von der Palette entfernt werden. Zusätzlich ermöglicht der OpenTOSCA Operation Invoker [22, 23] es Skripte direkt durch den OpenTOSCA-Container auszuführen, wodurch ein explizites Skript-Handling im Managementplan nicht weiter erforderlich ist.

Somit verbleiben TOSCA Node Management Tasks sowie TOSCA Topology Management Tasks als Grundlage für ein Modellierungskonzept welches im Folgenden vorgestellt wird. Grundsätzlich sind somit neben diesen, drei weitere Arten von Modellierungselementen zu unterscheiden: Start-Events, Gateways und End-Events. Im Rahmen dieser Arbeit werden lediglich AND-Gateways zur Modellierung berücksichtigt um den parallelen Ablauf von Aktivitäten abzubilden.

5.2.2 Modellierungskonzept des Datenflusses mit BPMN4TOSCA 2.0

Da eine Reduktion der (TOSCA) Data Objects im Managementplan einen großen Teil zur Vereinfachung der Modellierung beitragen kann, steht diese im Mittelpunkt des Konzeptes von BPMN4TOSCA 2.0. Nachdem bei der vorangehenden Betrachtung lediglich der Sequenzfluss, also die Abfolge der einzelnen Management Aktivitäten, betrachtet worden ist, wird im Folgenden die Modellierung des Datenflusses betrachtet.

Bei den Daten welche im Rahmen eines Managementplans produziert werden, handelt es sich um die Ein- und Ausgabeparameter von Aktivitäten, insbesondere von TOSCA Node- bzw. TOSCA Topology Management Tasks. Darüber hinaus verfügt ein Start-Event über Eingabeparameter und End-Events verfügen über Ausgabeparameter. Gateways steuern lediglich den Sequenzfluss und verfügen somit weder über Ein- noch Ausgabeparameter. Eingabeparameter können auf Daten, die vorher im Sequenzfluss produziert wurden, zugreifen. Entweder wird direkt auf Ausgabeparameter anderer Elemente zugegriffen oder es werden Verknüpfungsoperationen wie *concat* verwendet, um Ausgabeparameter miteinander zu konkatenieren. Da darüber hinaus keine Datentransformation vorgenommen wird, reicht der direkte Verweis auf die entsprechenden Ausgabeparameter bei dem Referenzieren von Eingabeparametern. Um den Zugriff auf die korrekten Daten sicherstellen zu können, werden diese auf Grundlage ihres eindeutigen Bezeichners im Managementplan referenziert, sodass Überschneidungen bei gleich benannten Ausgabeparametern vermieden werden können. Ferner werden lediglich Ausgabefelder von Aktivitäten angeboten welche im Sequenzfluss vor der aktuellen Aktivität modelliert und entsprechend über Sequenzflusskanten verbunden wurden. Ein konsistentes Verhalten bei der Modellierung wird sichergestellt, indem die Eingabeparameter des Start-Events gleichermaßen den Eingabeparametern des Managementplans entsprechen. Genauso werden die Ausgabeparameter des Managementplans in Form der Ausgabeparameter des End-Events modelliert.

Im vorherigen Beispiel entfällt somit die Notwendigkeit explizit TOSCA Data Objects modellieren zu müssen, um die entsprechenden Ausgabeparameter vorlagerter Management Aktivitäten zu lesen bzw. um diese in TOSCA Data Objects zu schreiben. Notwendige Eingabeparameter können somit direkt über den Bezeichner der Aktivität sowie den entsprechenden Ausgabeparameter referenziert werden.

Im Beispiel benötigen die TOSCA Node Management Tasks *InstallApacheWebserver* sowie *InstallMySQLServer* den Eingabeparameter *ipAddress*. Dieser wird als Ausgabeparameter *ipAddress* vom *CreateInstance* TOSCA Node Management Task bereitgestellt und im neuen Modellierungskonzept über *CreateInstance.ipAddress* referenziert.

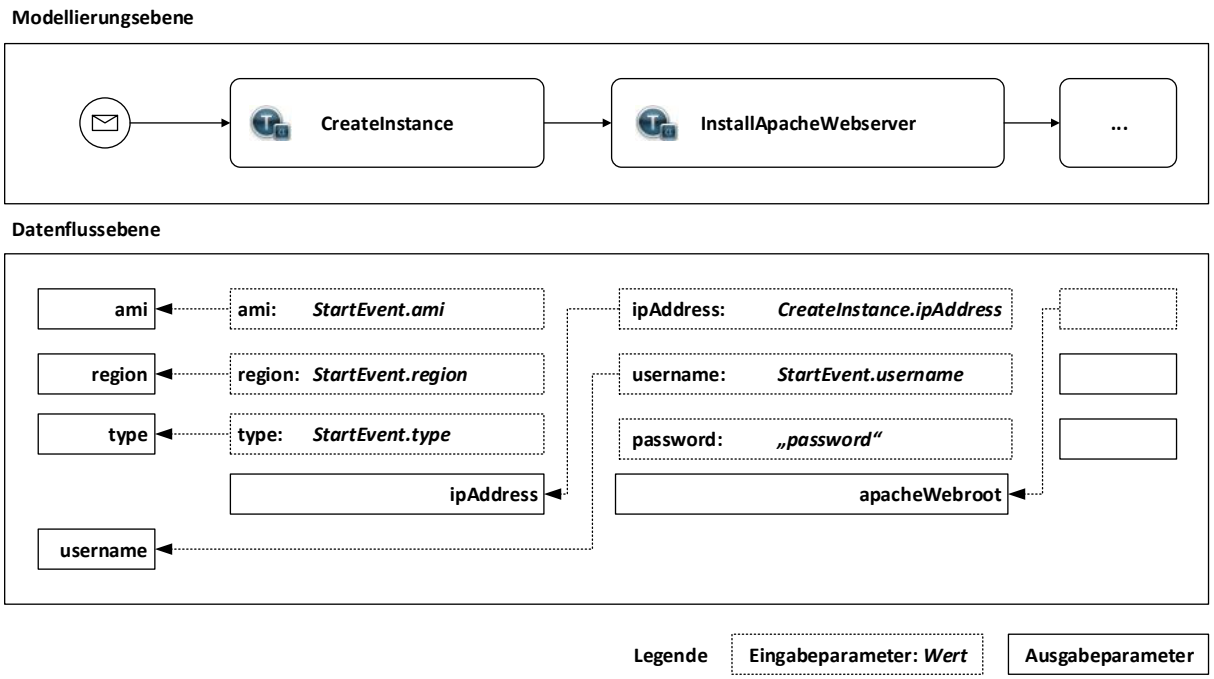

Abbildung 13: Referenzieren ohne explizite Modellierung von TOSCA Data Objects

Gleichermaßen können die Eingabeparameter *ami*, *region* sowie *type* des *CreateInstance* TOSCA Node Management Tasks entweder auf die Ausgabeparameter einer vorgelagerten Aktivität referenzieren oder beispielsweise auf die Eingabeparameter des Managementplans verweisen im dem entsprechend auf *StartEvent.ami*, *StartEvent.region* und *StartEvent.type* verwiesen wird. Als Ausgabeparameter des Managementplans ist die IP-Adresse der Amazon EC2 Instanz denkbar. Diese wird entsprechend als Eingabeparameter des End-Events definiert und verweist auf *CreateInstance.ipAddress*. Ein Ausschnitt der geschilderten Methode zum Referenzieren von Parametern wird in Abbildung 13 dargestellt.

5.3 Unterschiede zwischen BPMN4TOSCA und BPMN4TOSCA 2.0

Die neue domänenspezifische Sprache führt eine zusätzliche Abstraktionsebene im Gestaltungsprozess von Managementplänen ein. Hierdurch wird dieser Prozess, in Folge der Reduktion von BPMN4TOSCA, vereinfacht und ermöglicht eine intuitivere Erstellung von Managementplänen. BPMN4TOSCA 2.0 reduziert zudem BPMN4TOSCA um die Notwendigkeit explizit TOSCA Data Objects modellieren zu müssen, da diese nun explizit an den Sequenzfluss gebunden werden (siehe Abbildung 13). Der Managementplan spiegelt somit die reine Abfolge von Aktivitäten in Form von TOSCA Node Management Tasks sowie TOSCA Topology Management Tasks wieder. Die Kanten zwischen den Aktivitäten definieren über den Sequenzfluss hinaus im neuen Modellierungskonzept ebenfalls die Verfügbarkeit der vorgelagerten Ausgabeparameter für das Referenzieren in Eingabeparametern.

Diese Entwicklung fördert implizit stimmige Node Types zu entwickeln, die untereinander kompatibel sind. Man kann somit zu einer Baustein-artigen Modellierung übergehen, in der untereinander kompatible Bausteine zu einem Managementplan kombiniert werden und somit Fehler bei der Modellierung vermieden werden können. Ein ähnliches Verhalten wurde bereits in der Winery integriert, indem eine Verbindung von nicht kompatiblen Node Types durch entsprechende Relationship Types unterbunden wurde. Somit konnte mit BPMN4TOSCA 2.0 ein ähnlich implizit restriktives Verhalten wie in der Winery sichergestellt werden.

6. Modellierung von BPMN4TOSCA 2.0 Modellen

Bei BPMN4TOSCA 2.0 handelt es sich, genauso wie bei BPMN4TOSCA, um eine domänenspezifische Erweiterung von BPMN, die im Rahmen dieser Arbeit ausschließlich in einem entsprechenden Modellierungswerkzeug zur Verfügung steht und somit keine Veränderungen an der TOSCA-Laufzeitumgebung erfordert. In diesem Kapitel werden daher die Anforderungen für das zu entwickelnde Modellierungswerkzeug (im Folgenden: BPMN4TOSCA 2.0 Modeler) zusammengetragen sowie ein Bedienungskonzept für die Modellierung von BPMN4TOSCA 2.0 Modellen vorgestellt. Die Details der technischen Realisierung folgen im anschließenden Kapitel.

6.1 Anforderungen

Das neue Modellierungswerkzeug soll in das bestehende OpenTOSCA-Ökosystem integriert werden. Da es sich sowohl bei Winery (Abschnitt 2.4.3) als auch bei der Vinothek (Abschnitt 2.4.2) um browserbasierte Werkzeuge handelt, liegt es nahe das neue Modellierungswerkzeug ebenfalls für eine Nutzung im Browser verfügbar zu machen. Die browserbasierte Natur von Werkzeugen ermöglicht eine unkomplizierte Ausführung im Webbrowser ohne eine vorherige Installation auf dem Client notwendig zu machen. Neben der Nutzung des Modellierungswerkzeuges auf dem Desktop soll eine Bedienung auf touchfähigen Geräten möglich sein. Da es sich bei touchfähigen Geräten oft um Geräte mit eingeschränkter Funktionalität handelt, müssen zunächst die Unterschiede zwischen Desktop und touchfähigen Geräten in Betracht gezogen werden. Hierzu zählen insbesondere ein anderes Eingabemedium sowie eine mit Einschränkungen nutzbare Tastatur. So werden beispielsweise je nach eingesetztem Eingabemedium Ereignisse wie Click-Events auf touchfähigen Geräten durch Tap-Events abgebildet [25]. Andere auf dem Eingabemedium Maus basierende Ereignisse, wie *MouseMove* oder *MouseOver,* sind, je nach Gerät, überhaupt nicht oder nur eingeschränkt nutzbar [26]. Demnach muss im BPMN4TOSCA 2.0 Modeler eine entsprechende Abstraktion für unterschiedliche Eingabemedien vorhanden sein. Gleichermaßen gilt es die eingeschränkten Eingabemöglichkeiten durch nicht vorhandene Tasten auf den in touchfähigen Geräten eingesetzten Tastaturen zu berücksichtigen. Neben den Unterschieden in Bezug auf das Eingabemedium zeichnen sich touchfähige Geräte in der Regel durch eine geringere Bildschirmgröße aus. Somit gilt es den zur Verfügung stehenden Raum, auf verschiedenen Bildschirmgrößen, durch eine entsprechend gewählte Aufteilung der GUI optimal zu nutzen.

Durch die Berücksichtigung der geschilderten Unterschiede zwischen Desktop und touchfähigen Geräten kann mit Hilfe entsprechender Bibliotheken eine konsistente Benutzeroberfläche über verschiedene Gerätetypen hinweg realisiert werden. Diese Anforderungen wurden im folgenden Bedienkonzept berücksichtigt. Die konkrete technische Realisierung wird in Kapitel 7 thematisiert.

6.2 Bedienungskonzept

Der BPMN4TOSCA 2.0 Modeler besteht aus einer großen Zeichenfläche und einer Werkzeugleiste (siehe Abbildung 14). Diese enthält neben einer speziell für die Modellierung von Managementplänen optimierte BPMN-Palette die zuvor definierten BPMN4TOSCA 2.0 Elemente. Darüber hinaus verfügt die Werkzeugleiste über einen Button zum Speichern von Managementplänen. Im Gegensatz zur Topology Modeler GUI von Winery werden die Elemente der Werkzeugleiste des BPMN4TOSCA 2.0 Modelers horizontal im oberen Bereich der Zeichenfläche angeordnet, da anders als Topologien Managementpläne horizontal von links nach rechts modelliert werden und somit der zur Verfügung stehende Raum für die Zeichenfläche optimal genutzt werden kann.

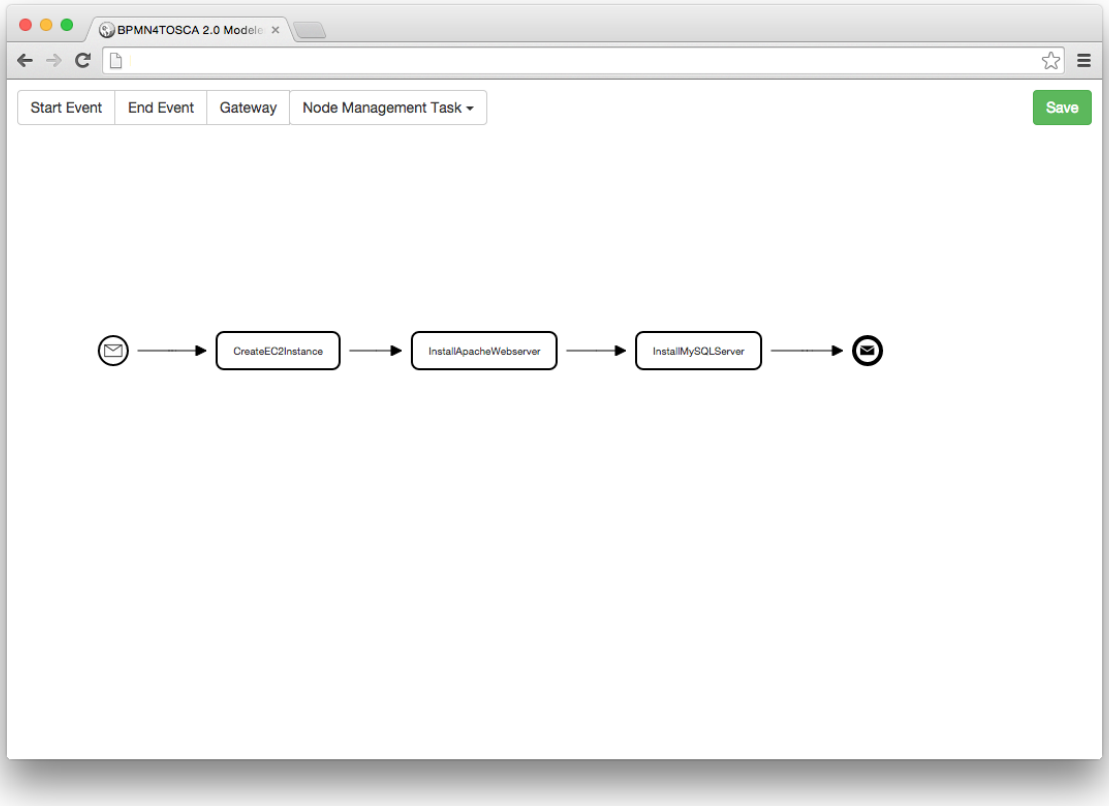

Abbildung 14: BPMN4TOSCA 2.0 Modeler Zeichenfläche

Die Elemente lassen sich per Drag-and-Drop von der Werkzeugleiste auf die Zeichenfläche bewegen auf der sie, ebenfalls per Drag-and-Drop, positioniert werden können. Hierbei wird darauf geachtet, dass die eingesetzten Elemente auf der Zeichenfläche, falls möglich, in mindestens einer Achse zueinander ausgerichtet werden. Um auf diesem Weg einen gut leserlichen Managementplan zu modellieren kommt hierbei ein dynamisch generiertes Ausrichtungsraster zum Einsatz.

Einzelne Elemente auf der Zeichenfläche lassen sich per Einfach-Klick (bzw. Tap auf touchfähigen Geräten) fokussieren. Fokussierte Elemente lassen sich mit Hilfe der Entfernen-Taste auf der Tastatur von der Zeichenfläche entfernen. Auf touchfähigen Geräten (die über keine entsprechende Taste zum Entfernen verfügen) kann das angezeigte Entfernen-Symbol genutzt werden. Darüber hinaus können einzelne Elemente im fokussierten Zustand per Drag-and-Drop vom vorangehenden zum nachfolgenden Element verbunden werden um somit den korrekten Sequenzfluss zu modellieren.

Per Doppelklick (bzw. DoubleTap auf touchfähigen Geräten) können die Eigenschaften von Elementen (mit Ausnahme von Gateways, die in diesem Sinne über keine Eigenschaften verfügen) bearbeitet werden. Hierzu wird ein separater Eigenschaften-Dialog geöffnet, in dem die Bezeichnung der Aktivität sowie optionale Ein- und Ausgabeparameter bearbeitet werden können. Zusätzlich werden in diesem Dialog ebenfalls das Node Template sowie die auszuführende Operation von TOSCA Node Management Tasks und auf diesem Weg die zugehörigen Ein- und Ausgabeparameter der gewählten Operation eingestellt.

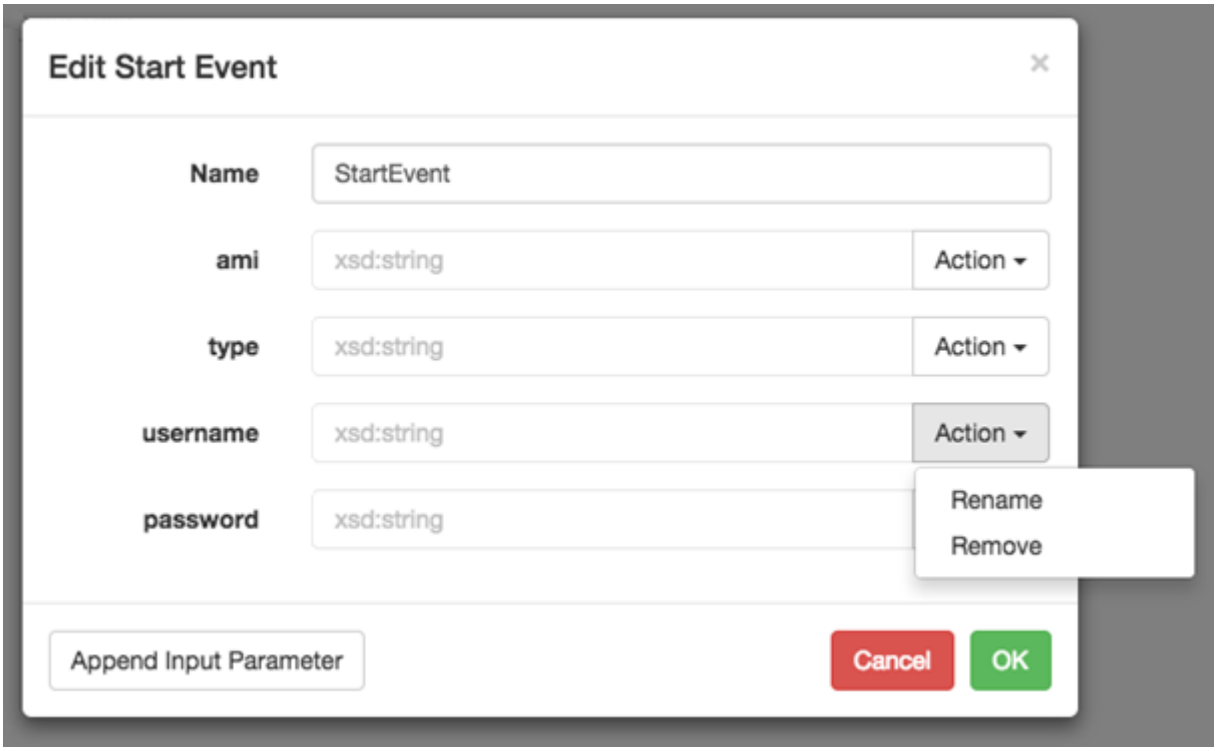

Abbildung 15: Bearbeitung der Eingabeparameter eines Managementplans im BPMN4TOSCA 2.0 Modeler

Jedes Element auf der Zeichenfläche wird über einen eindeutigen Bezeichner identifiziert, der in der Regel, in substantivierter Form die Aktivität, für die das Element steht, beschreibt. Zusätzlich verfügt jedes Element auf der Zeichenfläche (mit Ausnahme von Gateways) über Eingabe- oder Ausgabeparameter oder Eingabe- und Ausgabeparameter. Eingabeparameter beschreiben die für die Aktivität notwendigen Argumente. Ausgabeparameter beschreiben die Rückgabewerte von Aktivitäten. Die Eingabeparameter des kompletten Managementplans werden über den Eigenschaften-Dialog des Start-Events festgelegt, die Ausgabeparameter entsprechend im Dialog des End-Events. Anders als bei TOSCA Topology Management Tasks und TOSCA Node Management Tasks, deren Eingabe- bzw. Ausgabeparameter über die entsprechend ausgewählte Operation definiert sind, lassen sich in diesen beiden Ereignissen im Eigenschaften-Dialog neue Eingabe- bzw. Ausgabeparameter hinzufügen. Diese lassen sich ferner einzeln umbenennen oder löschen (siehe Abbildung 15).

Das Referenzieren von Eingabeparametern folgt dem im Abschnitt 5.2.2 erläuterten Modellierungskonzept des Datenflusses in BPMN4TOSCA 2.0. Sobald ein entsprechendes Eingabefeld fokussiert wird, werden die Ausgabeparameter vorgelagerter Aktivitäten in einem Auto-Complete-Dropdown zur Auswahl vorgeschlagen (siehe Abbildung 16). Neben dem direkten Referenzieren von Ausgabeparametern aus dem Managementplan ist es ferner möglich, eine feste Zeichenkette als Wert für einen Eingabeparameter zu hinterlegen, einen Wert aus der Topologie zu lesen oder den Verknüpfungsoperator *concat* zu nutzen um mehrere Werte miteinander zu verketten. Für die Auswahl dieser Optionen steht rechts neben dem Eingabefeld ein weiteres Dropdown zur Verfügung (siehe Abbildung 16).

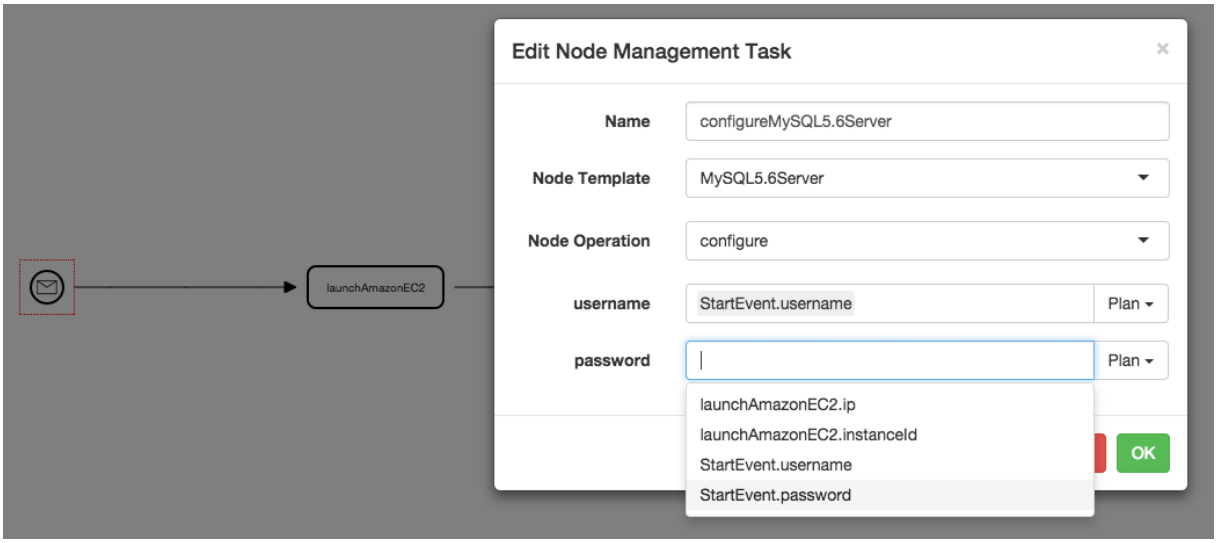

Abbildung 16: Referenzieren von Parametern im BPMN4TOSCA 2.0 Modeler

7. Architektur und technische Realisierung

Im Folgenden wird die Architektur des zuvor geschilderten BPMN4TOSCA 2.0 Modelers beschrieben sowie im Detail auf die technische Realisierung dieses Werkzeuges eingegangen. Bei der Realisierung wurden insbesondere die zuvor zusammengetragenen Anforderungen berücksichtigt um eine Modellierung von BPMN4TOSCA 2.0 auf verschiedenen Endgeräten innerhalb einer konsistenten GUI zu ermöglichen.

7.1 Architektur

Der BPMN4TOSCA 2.0 Modeler ist innerhalb des OpenTOSCA-Ökosystems wie in Abbildung 17 gezeigt eingebunden. Eine browserbasierten GUI ermöglicht die Modellierung von Managementplänen in BPMN4TOSCA 2.0. Hierbei kommen folglich clientseitige Technologien zum Einsatz. Das Speichern und Laden der modellierten Managementpläne erfolgt über die Winery Repository REST-Schnittstelle.

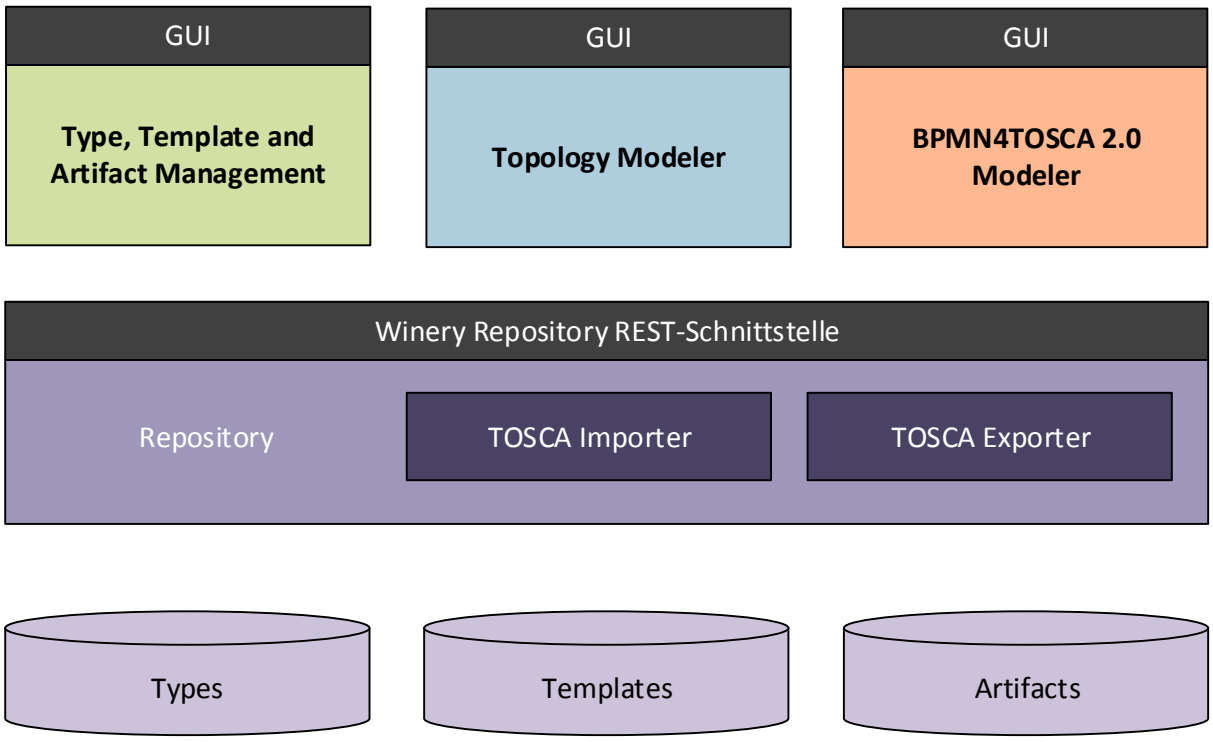

Abbildung 17: Integration des BPMN4TOSCA 2.0 Modelers im Winery-Umfeld (nach [27])

7.2 Integration im OpenTOSCA-Ökosystem

Managementpläne werden im OpenTOSCA-Ökosystem in Winery erzeugt und verwaltet. Um den BPMN4TOSCA 2.0 Modeler aufzurufen muss daher in der Winery das zugehörige Service Template ausgewählt werden um den BPMN4TOSCA 2.0 Modeler im richtigen Kontext aufzurufen. Dieser definiert die daraufhin im BPMN4TOSCA 2.0 Modeler verfügbaren Node Types für die Modellierung von TOSCA Node Management Tasks. Wie bereits im Abschnitt 2.4.3 erläutert, bietet Winery nicht nur eine grafische Benutzeroberfläche zur Organisation dieser Managementpläne, sondern stellt darüber hinaus eine REST-Schnittstelle zur Anbindung an das Repository bereit. Diese wird dazu genutzt die für die Modellierung notwendigen Informationen abzurufen. Hierbei handelt es sich u.a. um die im aktuellen Service Template verfügbaren Node Types, deren Properties sowie deren Interfaces inklusive der zugehörigen Operationen und Parameter die bei der Modellierung von TOSCA Node Management Tasks zur Verfügung stehen.

Das Laden und Speichern von Managementplänen erfolgt ebenfalls mit Hilfe der REST-Schnittstelle. Somit können Managementpläne mit Hilfe des BPMN4TOSCA 2.0 Modelers im Browser modelliert, anschließend serialisiert und über die REST-Schnittstelle im Repository der Winery gespeichert werden. Entsprechend können Managementpläne über die REST-Schnittstelle aus dem Repository der Winery geladen, deserialisiert und im BPMN4TOSCA 2.0 Modeler überarbeitet werden. Folglich muss im BPMN4TOSCA 2.0 Modeler selbst kein persistentes Verhalten integriert werden. Die Serialisierung erfolgt hierbei auf Grundlage von JavaScript Object Notation (JSON) [28]. Hierbei handelt es sich um ein Datenaustauschformat welches sich durch seine Einfachheit im Vergleich zur Auszeichungssprache XML auszeichnet und so mit einem geringeren Overhead auskommt [29]. Die generierte JSON-Zeichenkette besteht aus allen im aktuellen Managementplan eingesetzten Elementen. Jedes dieser Elemente wird beim Speichern einzeln serialisiert und bei Laden entsprechend deserialisiert um den Managementplan weiter bearbeiten zu können.

Der gespeicherte Managementplan muss somit anschließend lediglich entsprechend einer der im Abschnitt 2.7 erläuterten Vorgehensweisen in ausführbares BPEL umgewandelt werden. Der Fokus dieser Arbeit liegt jedoch ausschließlich in der Konzeption und Implementierung des Werkzeuges selbst, sodass dieser Schritt nicht betrachtet wird.

7.3 Technische Realisierung

Der BPMN4TOSCA 2.0 Modeler basiert auf der JavaScript Bibliothek Backbone.js (nachfolgend kurz: Backbone). Backbone hilft bei der Strukturierung von Anwendungen durch die Einführung von Models, Collections und Views als Teil einer Abwandlung des weit verbreiteten MVC-Patterns (Model-View-Controller). In diesem Fall werden jedoch weite Teile der Controller-Funktionalität in die Views ausgelagert. Diese Views sind somit für die Darstellung im Browser und die Verarbeitung von Benutzerinteraktion zuständig. Hierbei können Views beliebig geschachtelt sein, sodass unterschiedliche Views für die Darstellung und Verwaltung von verschiedenen Teilbereichen zuständig sind. Für die Anwendung relevanten Daten werden in Models gekapselt, welche die Zugriffe auf die Daten entsprechend abstrahieren und für die Validierung sowie Persistenz dieser Daten verantwortlich sind. Mehrere Models werden durch Collections zusammengefasst. Die Kommunikation zwischen diesen unterschiedlichen Komponenten einer Backbone-Anwendung erfolgt in der Regel mit Hilfe von Events, welche entsprechende Ereignisse gegenüber anderen Komponenten signalisieren.

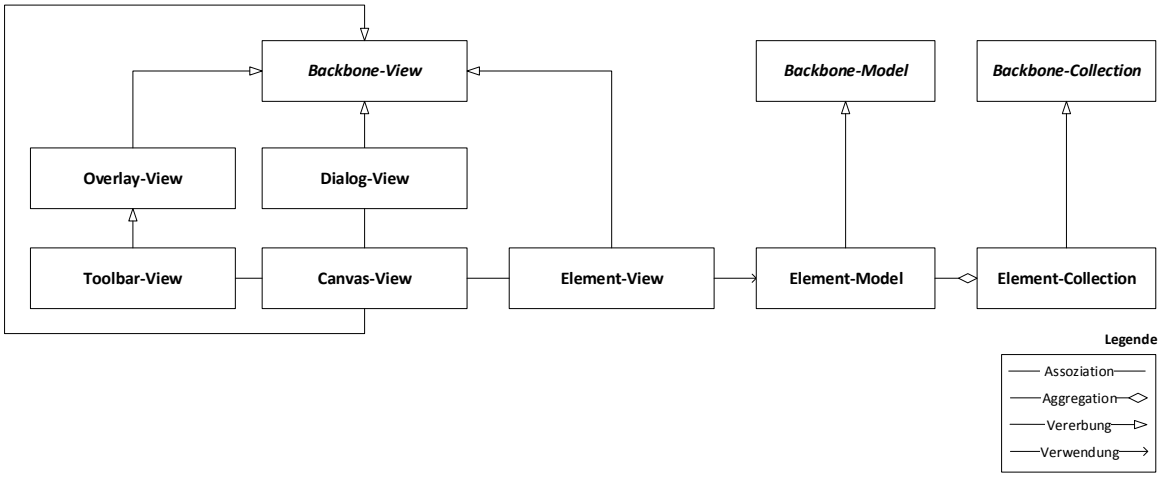

Abbildung 18: Komponenten des BPMN4TOSCA 2.0 Modelers

Gemäß dieser Vorgaben von Backbone wird ein Managementplan im BPMN4TOSCA 2.0 Modeler in Form einer Collection repräsentiert. Die einzelnen BPMN4TOSCA 2.0 Elemente sind folglich mit Hilfe von Models in dieser Collection organisiert. Eine spezielle Helper-Klasse ermöglicht die Anbindung an die Winery zum Laden und Speichern von Managementplänen. Die Zeichenfläche selbst, die Werkzeugleiste sowie sämtliche Dialoge werden mit Hilfe von Views dargestellt, welche für die Verarbeitung von Benutzerinteraktionen zuständig sind. Sämtliche BPMN4TOSCA 2.0 Elemente aus der Collection werden auf der Zeichenfläche durch entsprechende Views repräsentiert. Abbildung 18 zeigt die beschrieben Abhängigkeiten der einzelnen Komponenten welche im Folgenden im Detail erläutert werden.

Das Element-Model erbt von der Backbone-Model Klasse. Das Element-Model erweitert das Standard-Backbone-Model um spezielle Methoden zur Handhabung von Verbindungen zwischen verschieden Elementen in der Collection. Die einzelnen Element-Models werden in einer Element-Collection organisiert. Diese erbt von der Backbone-Collection und erweitert diese um einige Hilfsmethoden. Dazu zählen u.a. Methoden um die Collection in Form eines Graphen, ausgehend von einem Startknoten, zu traversieren. Jedes Element aus der Collection wird zudem auf der Zeichenfläche durch eine zugehörige View repräsentiert.

Die Zeichenfläche wird durch die Canvas-View-Komponente (nachfolgend kurz: Canvas) repräsentiert. Auf dem Canvas werden die einzelnen BPMN4TOSCA 2.0 Elemente durch einzelne Element-View-Komponenten repräsentiert. Der Canvas implementiert durch die jQuery UI Bibliothek zudem die Droppable-Eigenschaft, die es ermöglicht Element-Views per Drag-and-Drop auf dem Canvas zu platzieren. Darüber hinaus ist der Canvas für die Berechnung eines speziellen Hilfsrasters bei der Positionierung von Elementen auf den Canvas verantwortlich.

Auf dem Canvas selbst können zudem Erweiterungen in Form von Overlays registriert werden. Bei der Werkzeugleiste (auf der sämtliche BPMN4TOSCA-Elemente vertreten sind) handelt es sich um eine solche Overlay-View. Wird ein BPMN4TOSCA-Element per Drag-and-Drop von dieser Werkzeugleiste auf dem Canvas platziert, ist der Canvas für die Erstellung eines neuen Element-Views verantwortlich. Mit Hilfe von Overlay-Views lässt sich die GUI des BPMN4TOSCA 2.0 Modelers somit um neue Funktionen erweitern.

Die Element-View-Komponente (nachfolgend kurz: Element) repräsentiert ein BPMN4TOSCA-Element auf dem Canvas. Dieses Element wird an ein zugehöriges Element-Model gebunden. Mit Hilfe der jsPlumb-Bibliothek implementiert jedes Element eine erweiterte jQuery UI Draggable-Eigenschaft, welche über die reine Positionierung per Drag-and-Drop hinaus, die Möglichkeit eröffnet Elemente auf dem Canvas per Drag-and-Drop über eine Kante zu verbinden. Die Darstellung der unterschiedlichen Elemente erfolgt mit Hilfe der Formatierungssprache Cascading Style Sheets (CSS), die Darstellung der Kanten erfolgt mit Hilfe von Pfaden die mit Hilfe von Scalable Vector Graphics (SVG) dargestellt werden.

Sämtliche zur Modellierung verfügbaren Elemente erben von der o.g. Element Klasse. Zu diesem Zeitpunkt existieren Implementierung des Start-Events, des End-Events, des AND-Gateways, des TOSCA Node Management Tasks sowie des TOSCA Topology Management Tasks. Die einzelnen Elemente sind neben der visuellen Darstellung, durch das jeweilige BPMN- bzw. BPMN4TOSCA-Symbol, ebenfalls für die Steuerung des zugehörigen Eigenschaften-Dialogs verantwortlich. Diese Konfiguration der Elemente findet in den Klassen selbst statt. Neue BPMN-Elemente können somit hinzugefügt werden indem eine neue Kinderklasse von Element erzeugt wird in welcher die entsprechende Konfiguration vorgenommen wird.

Mit Hilfe der Dialog-View-Komponente (nachfolgend kurz: Dialog) können die Eigenschaften sämtlicher auf dem Managementplan verfügbarer Elemente bearbeitet werden. Der visuelle Rahmen des Dialogs wird hierbei durch die Modal-Komponente der Bootstrap-Bibliothek bereitgestellt. Der Dialog kapselt diese Komponente in einer Backbone-View um diese im Kontext des BPMN4TOSCA 2.0 Modelers strukturiert einbinden zu können. Beim Aufruf eines Dialogs wird das zu bearbeitende Element auf dem Canvas referenziert.

Die vorgenommenen Änderungen an den Eigenschaften dieses Elements werden anschließend unmittelbar an das zugehörige Element-Model propagiert. Zu den bearbeitbaren Eigenschaften in einem Dialog zählen neben dem Bezeichner sämtliche Ein- und Ausgabeparameter eines Elements. Innerhalb eines Dialogs können die Eingabeparameter mit Hilfe einer speziellen Auto-Complete-Komponente festgelegt werden, welche die Auswahl der entsprechenden Parameter erleichtert. Dieser basiert auf der Selectize-Komponente, für welche dynamisch zu jedem Zeitpunkt die korrekten Parameter zur Auswahl berechnet werden. Hierzu werden u.a. die von der Collection zur Verfügung gestellten Methoden zur Traversierung des Managementplans eingesetzt. Im Gegensatz zu den Eingabeparametern können die Ausgabeparameter, falls diese vom jeweiligen Dialog entsprechend implementiert sind, hinzugefügt, umbenannt oder gelöscht werden.

7.4 Technologieeinsatz

Da es sich bei der Winery um ein offizielles Produkt der Eclipse-Foundation handelt und der BPMN4TOSCA 2.0 Modeler in dieses Produkt integriert werden soll, wird darauf geachtet, dass sämtliche eingesetzten Komponenten und Bibliotheken den geforderten Code-Richtlinien und Lizenzbestimmungen gerecht werden. In diesem Fall handelt es sich insbesondere um die Apache License 2.0[10], die MIT-License[11] sowie die Eclipse Public License 1.0[12].

Beim BPMN4TOSCA 2.0 Modeler handelt es sich um ein browserbasiertes Werkzeug für die Modellierung von TOSCA-Managementplänen. Die Möglichkeit den BPMN4TOSCA 2.0 Modeler im Browser auszuführen, macht diesen folglich vom Einsatz moderner Webtechnologien im Browser abhängig. Um die Grundfunktionalität gewährleisten zu können basiert dieser auf den JavaScript Bibliotheken jQuery[13], jQuery UI[14], Backbone.js[15], Underscore.js[16] sowie auf den Komponenten jsPlumb[17], Selectize[18] und den User-Interface Komponenten von Bootstrap[19].

Darüber hinaus werden für die Unterstützung auf touchfähigen Geräten die Bibliotheken jQuery UI Touch Punch[20] sowie Tocca.js[21] benötigt um die zur Verfügung stehenden Eingabemöglichkeiten auf touchfähigen Geräten konsistent nutzen zu können.

[10] http://www.apache.org/licenses/LICENSE-2.0
[11] http://opensource.org/licenses/MIT
[12] http://www.eclipse.org/legal/epl-v10.html
[13] http://www.jquery.com
[14] http://www.jqueryui.com
[15] http://www.backbonejs.org
[16] http://www.underscorejs.org
[17] http://www.jsplumbtoolkit.com
[18] http://brianreavis.github.io/selectize.js
[19] http://www.getbootstrap.com
[20] http://touchpunch.furf.com
[21] http://gianlucaguarini.github.com/Tocca.js

8. Zusammenfassung und Ausblick

In dieser Arbeit wurde das Vorgehen bei der Modellierung eines TOSCA-Managementplans vorgestellt und kritisch reflektiert. Im Mittelpunkt dieser Untersuchung standen insbesondere die Betrachtung des OpenTOSCA-Ökosystems und dessen Defizite bei der Modellierung von TOSCA-Managementplänen. Basierend auf diesen Defiziten wurde mit BPMN4TOSCA 2.0 eine Erweiterung von BPMN4TOSCA konzipiert und vorgestellt, sowie ein entsprechendes Modellierungswerkzeug implementiert. Das Modellierungswerkzeug wurde darüber hinaus, durch eine Anbindung an die Winery, in das OpenTOSCA-Ökosystem integriert. Mit Hilfe der domänenspezfischen Sprache BPMN4TOSCA 2.0 sowie dem zugehörigen Modellierungswerkzeug ist es somit nun möglich sämtliche Managementaufgaben einer TOSCA-Anwendung in Anlehnung an eine zuvor modellierte Topologie zu erstellen. Wird eine der im Abschnitt 2.7 beschriebenen Transformationen zwischen BPMN4TOSCA 2.0 und WS-BPEL 2.0 im OpenTOSCA Container implementiert, kann mit OpenTOSCA eine End-to-End Open Source Toolchain für TOSCA angeboten werden. Diese Implementierung war im Rahmen dieser Arbeit jedoch nicht möglich.

Durch die erweiterbare Architektur des BPMN4TOSCA 2.0 Modelers ist es denkbar diesen in Zukunft um weitere Elemente der Standard BPMN-Palette zu ergänzen, um so die Möglichkeiten der Modellierung zu erweitern. Zu diesen Elementen zählen u.a. bedingte Abläufe für Kanten im Sequenzfluss oder die Möglichkeit Teilaufgaben zu definieren, die Aktivitäten gruppieren, um so den Managementplan noch verständlicher darzustellen. Darüber hinaus ist es denkbar, neben der Parallelität durch AND-Gateways weitere Formen von Verzweigungen im Sequenzfluss anzubieten, um beispielsweise ereignisbasierte Aktivitäten abzubilden.

BPMN4TOSCA 2.0 sowie der zugehörige BPMN4TOSCA 2.0 Modeler bieten somit ein breites Spektrum an Möglichkeiten zur Erweiterung an. Im Rahmen des OpenTOSCA-Ökosystems ist es denkbar die in Abschnitt 2.1.4 angedeutete Plan Generator Komponente um eine Generierung von Build-Plänen in BPMN4TOSCA 2.0 zu erweitern, sodass diese anschließend im BPMN4TOSCA 2.0 Modeler weiter individualisiert werden können. Ferner ist eine Erweiterung von BPMN4TOSCA 2.0 um eine Einbindung und Berücksichtigung von nicht-funktionalen Anforderungen denkbar. Eine Möglichkeit der Definition solcher Anforderungen wurde in Policy4TOSCA gezeigt. Jedoch muss hierfür vorab ein integriertes Konzept entwickelt werden, bevor eine konkrete Implementierung vorgenommen werden kann.

9. Literaturverzeichnis

[1] P. Mell und T. Grance, „The NIST Definition of Cloud Computing“, in *NIST Special Publication 800-145*, 2011.

[2] C. Shankar, V. Talwar, S. Iyer, Y. Chen, D. Milojicic und R. Campbell, „Specification-Enhanced Policies for Automated Management of Changes in IT Systems“, in *Proceedings of the 20th Conference on Systems Administration (LISA 2006)*, USENIX Association Berkeley, 2006.

[3] U. Breitenbücher, T. Binz, O. Kopp, F. Leymann und J. Wettinger, „Integrated Cloud Application Provisioning: Interconnecting Service-centric and Script-centric Management Technologies“, in *Proceedings of the 21st International Conference on Cooperative Information Systems (CoopIS 2013)*, Springer, 2013.

[4] T. Binz, U. Breitenbücher, O. Kopp und F. Leymann, „TOSCA: Portable Automated Deployment and Management of Cloud Applications“, in *Advanced Web Services*, Springer, 2014.

[5] T. Binz, G. Breiter, F. Leymann und T. Spatzier, „Portable Cloud Services Using TOSCA“, *IEEE Internet Computing, Bd. 16(03),* pp. 80-85, 2012.

[6] Organization for the Advancement of Structured Information Standards (OASIS), „Topology and Orchestration Specification for Cloud Applications Version 1.0“, 2013. [Online]. Available: http://docs.oasis-open.org/tosca/TOSCA/v1.0/TOSCA-v1.0.pdf.

[7] O. Kopp, T. Binz, U. Breitenbücher und F. Leymann, „BPMN4TOSCA: A Domain-Specific Language to Model Management Plans for Composite Applications“, in *Business Process Model and Notation*, Springer, 2012.

[8] Object Management Group (OMG), „Business Process Model And Notation, Version 2.0“, 2011. [Online]. Available: http://www.omg.org/spec/BPMN/2.0/PDF/.

[9] Organization for the Advancement of Structured Information Standards (OASIS), „Web Services Business Process Execution Language (WS-BPEL) Version 2.0“, 2007. [Online]. Available: http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.pdf.

[10] T. Binz, U. Breitenbücher, F. Haupt, O. Kopp, F. Leymann, A. Nowak und S. Wagner, „OpenTOSCA - A Runtime for TOSCA-based Cloud Applications“, in *11th International Conference on Service-Oriented Computing*, Springer, 2013.

[11] F. Leymann und D. Roller, „Production Workflow: Concepts and Techniques“, Prentice Hall, 1999.

[12] D. Chappell, „What is application lifecycle management?“, 2008. [Online]. Available: http://www.davidchappell.com/WhatIsALM--Chappell.pdf.

[13] M. Fetzer, „*Ein Marktplatz für TOSCA*“, Bachelorarbeit Nr. 65, Universität Stuttgart Institut für Parallele und Verteilte Systeme, 2013.

[14] T. Waizenegger, M. Wieland, T. Binz, U. Breitenbücher, F. Haupt, O. Kopp, F. Leymann, B. Mitschang, A. Nowak und S. Wagner, „Policy4TOSCA: A Policy-Aware Cloud Service“, in *On the Move to Meaningful Internet Systems: OTM 2013 Conferences*, Springer, 2013.

[15] U. Breitenbücher, T. Binz, K. Képes, O. Kopp, F. Leymann und J. Wettinger, „Combining Declarative and Imperative Cloud Application Provisioning based on TOSCA“, in *Proceedings of the IEEE International Conference on Cloud Engineering (IEEE IC2E 2014)*, IEEE Computer Society, 2014.

[16] O. Kopp, D.Martin, D. Wutke und F. Leymann, „The Difference Between Graph- Based and Block-Structured Business Process Modelling Languages“, in *Enterprise Modelling and Information Systems*, Gesellschaft für Informatik e.V. (GI), 2009, pp. 3-13.

[17] F. Leymann, „BPEL vs. BPMN 2.0: Should You Care?“, in *Business Process Modeling Notation*, Springer, 2010.

[18] O. Kopp, T. Binz, U. Breitenbücher und F. Leymann, „Winery – A Modeling Tool for TOSCA-based Cloud Applications“, in *11th International Conference on Service-Oriented Computing*, Springer, 2013.

[19] U. Breitenbücher, T. Binz, O. Kopp und F. Leymann, „Vinothek - A Self-Service Portal for TOSCA“, in *Proceedings of the 6th Central-European Workshop on Services and their Composition (ZEUS 2014)*, 2014.

[20] „OpenTOSCA - Open Source TOSCA Ecosystem“, [Online]. Available: http://www.iaas.uni-stuttgart.de/OpenTOSCA/.

[21] J. Mendling, K. B. Lassen und U. Zdun, „On the Transformation of Control Flow between Block-Oriented and Graph-Oriented Process Modeling Languages“, in *International Journal of Business Process Integration and Management (IJBPIM)*, 2006.

[22] J. Wettinger, T. Binz, U. Breitenbücher, O. Kopp und F. Leymann, „Streamlining Cloud Management Automation by Unifying the Invocation of Scripts and Services Based on TOSCA“, in *International Journal of Organizational and Collective Intelligence (IJOCI)*, IGI Global, 2014, pp. 45-63.

[23] J. Wettinger, T. Binz, U. Breitenbücher, O. Kopp, F. Leymann und M. Zimmermann, „Unified Invocation of Scripts and Services for Provisioning, Deployment, and Management of Cloud Applications Based on TOSCA“, in *Proceedings of the 4th International Conference on Cloud Computing and Services Science (CLOSER 2014)*, SciTePress, 2014.

[24] D. Moody, „The ‚Physics' of Notations: Towards a Scientific Basis for Constructing Visual Notations in Software Engineering", in *IEEE Transactions on Software Engineering , Bd. 35(6)*, IEEE, 2009, pp. 756-779.

[25] J. Bender, „Tap vs. Click: Death by Ignorance", 2012. [Online]. Available: https://coderwall.com/p/bdxjzg/tap-vs-click-death-by-ignorance.

[26] C. Wilson und P. Kinlan, „Touch And Mouse Together Again For The First Time", 2013. [Online]. Available: http://www.html5rocks.com/en/mobile/touchandmouse/.

[27] „Winery," Eclipse, [Online]. Available: http://projects.eclipse.org/projects/soa.winery.

[28] European Computer Manufacturers Association (ECMA), „The JSON Data Interchange Format", 2013. [Online]. Available: http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf.

[29] D. Crockford, „JSON: The Fat-Free Alternative to XML", in *The Proceedings of XML 2006*, Boston, 2006.

Alle Links wurden zuletzt am 05.01.2015 geprüft.

Erklärung

Hiermit versichere ich, diese Arbeit selbständig verfasst und nur die angegebenen Quellen benutzt zu haben. Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

_____

(Thomas Michelbach)