Oracle Database Appliance – ein Deep Dive

David Hueber, dbi services

Im Jahr 2011 wurde die Oracle Database Appliance als neues Engineered System eingeführt. Die für Datenbanken-Konsolidierung und Hochverfügbarkeit gedachte Plattform ist so gut wie eine „Plug and Play"-Lösung. Dieser Artikel gibt einen Überblick über die ODA-Architektur und -Basisnutzung sowie Schlüssel, um die Black Box zu umgehen und von der vollen Leistung der Lösung zu profitieren.

Die Oracle Database Appliance (ODA) gehört dem Portfolio der Engineered Systems an. Von einer „low-cost Exadata" ist definitiv nicht die Rede, denn Exadata steht für höchste Performance und Volumen. ODA wurde vielmehr als einfache, zuverlässige und verfügbare Lösung zur Konsolidierung von Datenbanken entwickelt.

Das System ist auf einer vollredundanten Hardware aufgebaut. Sie beinhaltet ein neues Framework, um Einrichtung und Management von Datenbanken zu vereinfachen, sowie auch einige (aber nicht alle) Technologien zur Hochverfügbarkeit wie beispielsweise RAC One Node und RAC. Außerdem ist die ODA die erste Plattform mit einem „Pay as you grow"-Lizenzmodell für Oracle-Datenbanken.

Die ODA-Architektur

Seit ihrer Version X3-2 (Q3 2013) ist die Oracle Database Appliance auf drei individuellen Komponenten aufgebaut: zwei Server und ein Speicher-Array *(siehe Abbildung 1)*. In ihrer letzten Version (ODA X4-2) enthält jeder Server:

- 2 Intel Xeon E5-2697v2 12 Cores
- 256 GB Memory
- 2 HDD SAS 10k 600 GB (gespiegelt in RAID 1, für „boot" und Binaries benutzt)
- 2 SAS HBA Adapters
- 2 Twinnax 10 GB Interconnect
- 4 Ethernet 100 MB Ports

Das Speicherarray beinhaltet 24 Disks, wie folgt verteilt:

- 20 HDD SAS 10k 900 GB (benutzt für Data und Recovery Files)
- 4 HDD SSD 200 GB – (benutzt für Redo Logs und Control Files)

Die komplette Hardware-Architektur von ODA, einschließlich Lüfter und Netzteile, ist voll redundant. Der Speicherzugriff erfolgt durch zwei getrennte Controller, die

Abbildung1: ODA-Frontplatten-Ansicht

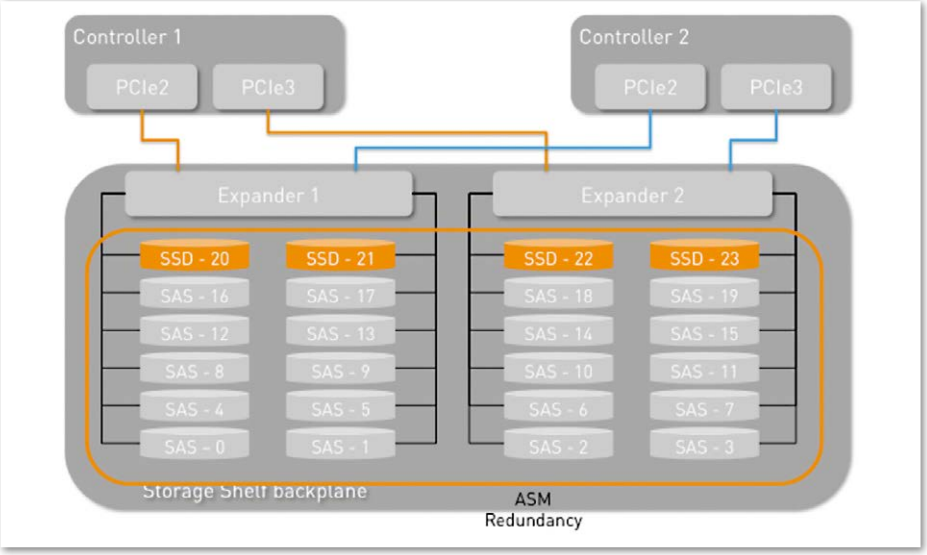

Abbildung 2: ODA-Speicherzugriff-Architektur

mit zwei Expandern innerhalb des Speicherarrays verknüpft sind. Die Daten-Redundanz erfolgt durch Oracle Automatic Storage Management (ASM). Bei Ausfall einer der Komponenten wird der Speicherzugriff weiter gewährt *(siehe Abbildung 2)*. Hinsichtlich der Speicherkapazität werden zwei Parameter berücksichtigt:

- ASM Redundancy
- Backup storage location

Es steht zur Wahl, entweder den Modus „Normal Redundancy" oder „High Redundancy" vom ASM zu nutzen, also entweder die Originalspeicher-Kapazität von 9 TB bis 18 TB (Normal Redundancy) oder darunter bis 6 TB (High Redundancy). Zum anderen wird die Speicherkapazität direkt davon beeinflusst, ob das Backup der Datenbanken auf der ODA selbst gespeichert ist (in der Fast Recovery Area) oder auf einem externen Speicherort wie zum Beispiel Tapes oder NFS-Shares. Je nach Wahl wird das ASM-Disk-Group-Sizing angepasst *(siehe Tabelle 1)*.

Demnach wird festgestellt, dass die ODA-Speicherkapazität für Oracle-Datenbanken je nach Einstellung von Redundanz und Backup-Ort von 2,4 TB bis 7,2 TB schwankt. Anmerkung: Diese Einstellungen werden während der ODA-Einrichtung konfiguriert und lassen sich im Nachhinein ohne Neuinstallation der ODA nicht mehr ändern.

Backup Modus	+DATA (TB)	+RECO (TB)	+REDO (GB)
Normal Redundancy			
Local Backup	3,6	4,5	248
External Backup	7,2	0,98	248
High Redundancy			
Local Backup	2,4	3	248
External Backup	4,8	0,65	248

Tabelle 1

Die Software-Architektur

Grundsätzlich kann jede ODA seit Version 2.5 nach zwei verschiedenen Architekturtypen installiert werden:

- Bare Metal
- Virtualized

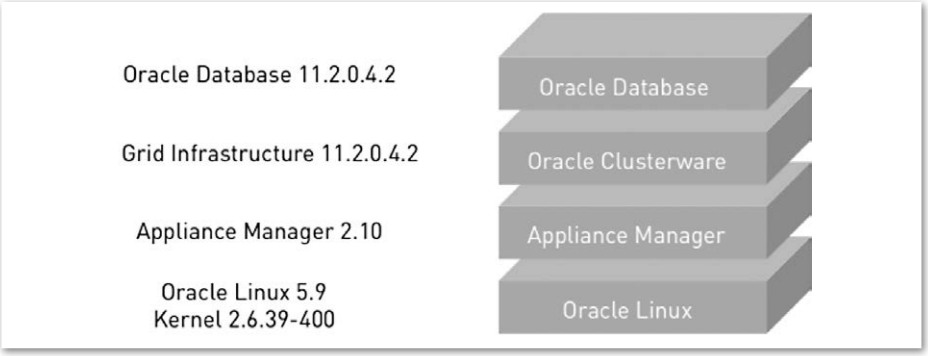

Abbildung 3: ODA – Bare-Metal-Software-Architekturtyp

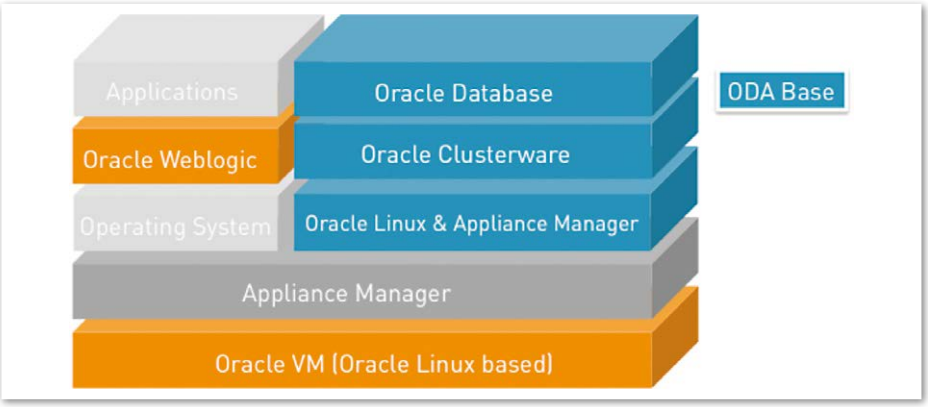

Abbildung 4: ODA – virtuelle Software-Architektur

Die Version 1.0 von ODA wurde mit dem Architekturtyp „Bare Metal" entwickelt. Dieser ist der Datenbank ausschließlich vorbehalten. Er basiert auf Oracle Linux und beinhaltet Grid Infrastructure sowie Oracle Database *(siehe Abbildung 3)*. Der in Version 2.5 eingeführte virtuelle Modus ermöglicht bei ODA das Betreiben von Oracle VM und infolgedessen von virtuellen Maschinen für Applikationen wie WebLogic, JD Edwards oder Tomcat neben einem virtuellen Server für Datenbanken *(siehe Abbildung 4)*.

Bei dieser Architektur laufen alle Datenbanken in einer einzigen und dedizierten virtuellen Umgebung, nämlich im ODA BASE. Je nach Bedarf an Software-Komponenten können daneben zusätzliche virtuelle Maschinen eingerichtet werden. Die virtuelle Maschine ODA BASE hat innerhalb der gesamten Plattform die führende Funktion und wird zur Verwaltung weiterer Komponenten benutzt:

- Datenbanken
- Binaries
- virtuelle Maschinen

Verschiedene Templates stehen für Applikationen virtueller Maschinen bereits zum Herunterladen unter „edelivery.oracle.com/linux" zur Verfügung. Zudem liefert Oracle eine WebLogic-Anwendung für ODA mit Java-basiertem Installationsprogramm, das eine hochverfügbare WebLogic-Struktur einschließlich Traffic Director automatisch aufstellt *(siehe „http://www.oracle.com/technetwork/middleware/weblogic-oda/downloads/index.html" und Abbildung 5)*.

Der Appliance Manager

Für Oracle-DBAs ist die größte Veränderung bei der ODA der Appliance Manager

„OAKCLI“, ein neues Command Line Interface, das folgende Funktionen ermöglicht:

- Hardware-Komponente auflisten
- Hardware-Komponente validieren und diagnostizieren
- Software installieren und upgraden
- Patches anwenden
- Datenbanken erstellen und entfernen
- Oracle Homes installieren und entfernen
- Virtuelle Maschinen einrichten und verwalten

Der OAKCLI-Befehl soll immer als Root-Benutzer vom ersten Knoten gestartet werden („node 0“), sonst gibt es eine Fehlermeldung *(siehe Listing 1)*. *Listing 2* zeigt die Festplattenüberprüfung mit OAKCLI und *Listing 3* die Validierung der Speicherstruktur und Verkabelung.

Verwaltung von Datenbanken unter Appliance Manager

Sobald die ODA konfiguriert wurde (Netzwerk, Hostname, Software-Einrichtung), können Datenbanken erstellt werden. Dies wird über den Appliance Manager realisiert, es muss also als „root“-Benutzer vom Knoten 0 gestartet werden *(siehe Listing 4)*. Beim Erstellen von Datenbanken werden verschiedene Parameter bestimmt:

- Name der Datenbank (obligatorisch)
- Version der Datenbank
- Datenbank-„NLS”-Einstellungen

Der CREATE-Befehl startet einen Installationsassistenten, der Basis-Informationen wie „root“-, „oracle“- und „sysasm“-Benutzerkennwort benötigt, aber auch den Typ von Datenbanken *(siehe Listing 5)* sowie die „Database Class“ *(siehe Listing 6)*.

Die „Database Class“ definiert in der Tat, welches „dbca template“ für das Erstellen der Datenbank benutzt werden soll. ODA wird mit acht Templates geliefert; weitere Templates sind nicht möglich *(siehe Tabelle 2)*.

Diese Einstellungen werden nur der Information halber angegeben, insbesondere die Datenbanken-Anzahl oder die IOPS. In den nächsten Schritten geht es dann darum, die Umgebung anzupassen.

Über die Limits hinaus

Die ODA bietet auf jeden Fall eine schnelle und einfache Methode, Datenbanken

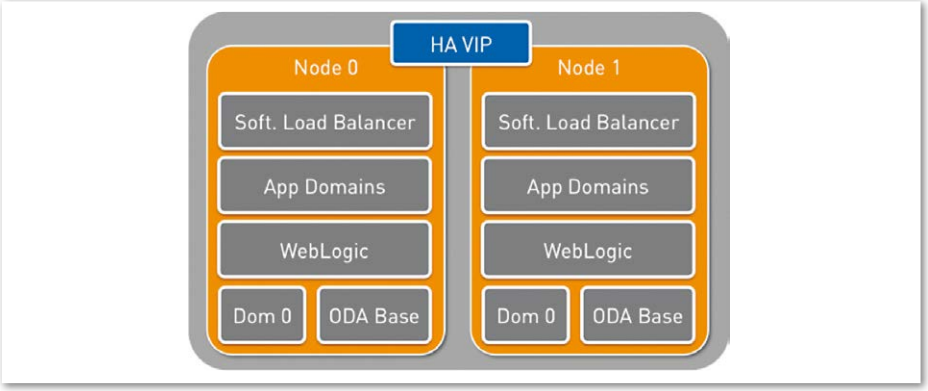

Abbildung 5: WebLogic-Anwendung für ODA

```
[oracle@dbi-oda2 ~]$ oakcli create database -d DBITEST2
sh: dmidecode: command not found
ERROR: 2014-01-06 22:53:25: Insufficient privileges to create the database

[root@dbi-oda2 ~]# oakcli create database -d DBITEST2
ERROR: 2014-01-06 22:54:18: oakcli create database should be invoked
from the first node
```

Listing 1

```
[root@dbi-oda1 ~]# oakcli show disk
   NAME            PATH            TYPE    STATE          STATE_DETAILS

   e0_pd_00        /dev/sda        HDD     ONLINE         Good
   e0_pd_01        /dev/sdb        HDD     ONLINE         Good
   e0_pd_02        /dev/sdaa       HDD     FAILED         DiskRemoved
   e0_pd_03        /dev/sdab       HDD     ONLINE         Good
   e0_pd_04        /dev/sdac       HDD     ONLINE         Good
```

Listing 2

```
root@dbi-oda2 ~]# oakcli validate -c storagetopology

 It may take a while. Please wait...
 INFO     : ODA Topology Verification
 INFO     : Running on Node1
 INFO     : Check hardware type
 SUCCESS  : Type of hardware found : X3-2
 INFO     : Check for Environment(Bare Metal or Virtual Machine)
 SUCCESS  : Type of environment found : Bare Metal
 INFO     : Check number of Controllers
 SUCCESS  : Number of Internal LSI SAS controller found : 1
 SUCCESS  : Number of External LSI SAS controller found : 2
 INFO     : Check for Controllers correct PCIe slot address
 SUCCESS  : Internal LSI SAS controller    : 50:00.0
 SUCCESS  : External LSI SAS controller 0  : 30:00.0
 SUCCESS  : External LSI SAS controller 1  : 40:00.0
 INFO     : Check if JBOD powered on
 SUCCESS  : 0JBOD : Powered-on
 INFO     : Check for correct number of EBODS(2 or 4)
 FAILURE  : Check for correct number of EBODS(2 or 4) : 1
 ERROR    : 1 EBOD found on system, which is less than 2 EBODS with 1
JBOD
 INFO     : Above details can also be found in log file=/
opt/oracle/oak/log/dbi-oda2/storagetopology/StorageTopolo-
gy-2014-01-07-23:28:52_16620_29386.log
```

Listing 3

```
[root@ODADBI1-base ~]# oakcli create database -db TTOP1 -version
11.2.0.3.10 -params top1conf
```

Listing 4

```
Please select one of the following for Database Deployment [1 .. 3]:
1      => EE : Enterprise Edition
2      => RACONE
3      => RAC
1
Selected value is: EE
Please select one of the following for Node Number [1 .. 2]:
1      => ODADBI1-base
2  => ODADBI2-base
2
Selected value is: ODADBI2-base
```

Listing 5

```
Specify the Database Class (1. Medium 2. Others) [1]:2

Please select one of the following for Database Class [1 .. 8] :
1      => Very Very Small
2      => Very Small
3      => Small
4      => Medium
5      => Large
6      => Extra Large
7      => Extra Extra Large
8      => Extra Extra Extra Large
3
Selected value is: Small
```

Listing 6

```
[root@ODADBI1-base ~]# oakcli show db_config_params -detail
Available DB configuration files are:
default
DATABASE_BLOCK_SIZE      =>      8192
DATABASE_LANGUAGE        =>      AMERICAN
DATABASE_CHARACTERSET    =>      AL32UTF8
DATABASE_TERRITORY       =>      AMERICA
COMPONENT_LANGUAGES      =>      en
```

Listing 7

in einer hochverfügbaren und effizienten Umgebung zum Laufen zu bringen. Da jede Anwendung unterschiedlich sein kann, können gewisse Anpassungen doch notwendig sein. Dafür lässt sich die Datenbank-Einstellung unter „OAKCLI" vom DBA individuell verbessern.

Als Erstes nehmen wir die NLS-Einstellungen der Datenbank unter die Lupe. Diese werden von einer Konfigurationsdatei von OAKCLI während der Datenbank-Erstellung definiert. Eine Ausgangskonfiguration wird hierfür zur Verfügung gestellt *(siehe Listing 7)*. Sollten andere NLS-Einstellungen nötig sein, kann individuell eine Konfigurationsdatei erzeugt und während der Datenbank-Erstellung verwendet werden *(siehe Listing 8)*. Nach Erstellung der Datenbank sollten verschiedene Parameter überprüft werden, etwa die Einstellungen für „control files" und „redo logs" *(siehe Listing 9)*.

Der Autor macht darauf aufmerksam, dass auch, wenn beide in der +REDO Disk Group (SSD Disks) gespeichert sind, weder der Control File noch die Redo Logs gespiegelt werden. Als Schutz gegen logische Korruption empfiehlt es sich, einen zusätzlichen Control File und einen Redo Member pro Redo Log Group hinzuzufügen. Dies sollte allerdings in der +REDO Disk Group angebracht werden, um Leistungseinbußen zu vermeiden.

Als Nächstes soll der Arbeitsspeicher-Verbrauch der Datenbank beobachtet werden. Seit Oracle 11*g* wurde das Automatic Memory Management (AMM – „memory_target") eingeführt. Dieses ist allerdings nicht mit „huge pages" kompatibel, daher nutzt ODA das Automatic Shared Memory Management (ASMM – „sga_target & pga_aggregate_target"). Standardmäßig kommt jede ODA aus dem Werk

Setting	XXS	XS	S	M	L	XL	XXL	XXXL
CPU_COUNT	2	2	4	8	12	24	32	48
SGA (GB)	2	4	8	16	24	48	63	96
PGA (GB)	1	2	4	8	12	24	32	48
Processes	200	200	400	800	1200	2400	3200	4800
Redo (GB)	1	1	1	2	4	4	4	4
Nb DBs	24	24	12	6	4	2	1	1
IOPS	137	137	275	550	825	1650	3300	3300

Tabelle 2

mit einer „huge pages“-Einstellung von 50 Prozent des physischen Arbeitsspeichers des ODA Servers *(siehe Listing 10)*. Wer über die volle Leistung des Arbeitsspeichers auf einer ODA verfügen möchte, folgt der MOS-Note „401749.1“, um die Größe der „huge page” zu erweitern.

Sobald die Datenbanken erstellt und angepasst sind, können Prinzipen zur Hochverfügbarkeit eingeführt werden. Hierfür stellt ODA „out of the box“ entweder RAC-One-Node- oder RAC-Lösungen zur Verfügung. Diese Optionen sind unter der Entreprise Edition kostenpflichtig. Folgende Optionen sind ebenfalls verfügbar:

- Data Guard (in der Enterprise Edition beinhaltet)
- Failover Cluster (in der Grid Infrastructure beinhaltet)

Während Data Guard eine zweite ODA benötigt, kann Failover Cluster mit einem einzigen Befehl als „root“-Benutzer implementiert werden *(siehe Listing 11)*. Dann kann die Instanz von einem Knoten zum anderen gewechselt werden *(siehe Listing 12)*.

Lizenzierung

Der Vollständigkeit halber möchte der Autor noch ein kurzes Wort zur Lizenzierung hinzufügen, ein spannendes Thema bei Oracle-Umgebungen. ODA ist die erste Plattform mit „Pay-as-you-grow“-Lizenzmodell. Bei Auslieferung ist sie für den Standard-Oracle-Prozessor lizenziert inklusive aller Regeln bezüglich Intel-Prozessoren (wie Core Factor). Dennoch besteht ein relevanter Unterschied zu den auf fremder Hardware basierenden Umgebungen (HP, IBM, DELL etc.), bei denen nicht alle verfügbaren Cores zwangsläufig lizenziert werden müssen. Von 24 verfügbaren Cores kann mit einer Minimum-Anzahl gestartet und von da an je nach Bedarf weitere Cores freigeschaltet werden. Aufstocken geht, reduzieren allerdings nicht. In diesem Zusammenhang besteht ein Unterschied zwischen der Bare-Metal- und der virtuellen Architektur, da die Mindestanzahl der zu lizenzieren Cores abweicht:

- *Bare Metal*
  Minimum von vier Cores pro Knoten, Erweiterung in Stufen von jeweils vier zusätzlichen Cores pro Knoten

```
[root@ODADBI1-base ~]# oakcli create db_config_params -conf dbprod1
Please select one of the following for Database Block Size [1 .. 4]:
1     => 4096
2     => 8192
3     => 16384
4     => 32768
2
Selected value is: 8192

Specify the Database Language (1. AMERICAN 2. Others) [1]:1
Selected value is: AMERICAN

Specify the Database Characterset (1. AL32UTF8 2. Others) [1]:2

Please select one of the following for Database Characterset [0 .. 10] :
0     => Others
1     => AL32UTF8
...
...
90    => UTF8
...
96    => WE8ISO8859P15
97    => WE8ISO8859P9
98    => WE8MACROMAN8S
96
Selected value is: WE8ISO8859P15

Specify the Database Territory (1. AMERICA 2. Others) [1]:1
Selected value is: AMERICA

Specify the Component Language (1. en 2. Others) [1]:1
Selected value is: en

Successfully generated the Database parameter file 'top1conf'

[root@ODADBI1-base ~]# oakcli create database -db DBPROD -params dbprod
```

Listing 8

```
SQL> show parameter control_files
NAME              TYPE        VALUE
------------------- --------   ------------------------------
control_files     string +REDO/dbitest/control01.ctl

SQL> select group#,members from v$log;
GROUP#MEMBERS
-------   ----------
1     1
2     1
3     1
```

Listing 9

```
[root@dbi-oda1 ~]# grep -i huge /proc/meminfo
HugePages_Total:      64000
HugePages_Free:       56200
HugePages_Rsvd:       393
HugePages_Surp:       0
Hugepagesize:         2048 kB
```

Listing 10

```
[root@ODADBI2-base bin]# ./crsctl modify res ora.dbprod.db -attr
"PLACEMENT=favored"
```

Listing 11

```
[root@ODADBI2-base bin]# ./crsctl relocate res ora.dbprod.db
CRS-2673: Attempting to stop 'ora.dbprod.db' on 'ODADBI2-base'
CRS-2677: Stop of 'ora.dbprod.db' on 'ODADBI2-base' succeeded
CRS-2672: Attempting to start 'ora.dbprod.db' on 'ODADBI1-base'
CRS-2676: Start of 'ora.dbprod.db' on 'ODADBI1-base' succeeded
```

Listing 12

David Hueber
david.hueber@dbi-services.com

- *Virtuelle Architektur*
  Minimum von zwei Cores pro Knoten, Erweiterung in Stufen von jeweils zwei zusätzlichen Cores pro Knoten

Fazit

ODA ist eine ziemlich einfache, zuverlässige und verfügbare Plattform für das Konsolidieren von Oracle-Datenbanken. Sie bietet eine intuitive und integrierte Methode zu deren Erstellung und Verwaltung. Nichtdestotrotz muss hinter die Kulissen geschaut werden, um zu sehen, wie die ODA funktioniert und wie sie anzupassen ist, um so viele Vorteile wie möglich zu erzielen.

Inventur im ERP-System

Christian Wille, PITSS GmbH

Oracle-Forms-Anwendungen haben in der Regel eine lange Historie hinter sich und viele Änderungen durchlaufen – umgesetzt von unterschiedlichen Entwicklern, sodass meist ein kaum überschaubares und nur noch schwer zu beherrschendes System übrig geblieben ist. Nur selten werden Qualitätsvorgaben berücksichtigt, die aufgrund ihrer Verwendung für mehr Klarheit sorgen. Ein Kundenbeispiel zeigt, wie man an diese Problematiken herangehen kann, um eine solch gewachsene Anwendung wieder wartbar und beherrschbar zu machen.

Ein deutsches Handelsunternehmen kaufte bei seiner Gründung im Jahr 2004 ein ERP-System eines anderen Unternehmens, um seine Geschäftsprozesse abbilden zu können. Analog zu der rasanten Unternehmensentwicklung auf mehr als 1.400 Filialen in fünf Ländern (Stand 2014) wuchs und veränderte sich auch das ERP-System. Zahlreiche Prozesse wurden optimiert, neue Prozesse geschaffen und alte abgeschaltet. Auch der Ersatz einzelner Komponenten, etwa des Logistiksystems oder der Verwaltung der Mietobjekte, durch Standardsoftware hatte große Auswirkungen auf das System, da diverse Objekte nicht mehr benötigt und durch neue Schnittstellen ersetzt wurden.

Ein solcher Produkt-Lifecycle ist bei Forms-Anwendungen durchaus verbreitet. Es gelingt in der Regel so gut wie nie, die nicht benötigten Komponenten aus dem System zu entfernen und die zunehmende Komplexität herauszuhalten. Das Unternehmen beauftragte eine detaillierte Code-Analyse, um einerseits einen Überblick über das bestehende System zu erhalten und andererseits nicht mehr benötigte Komponenten aus der Anwendung entfernen zu können.

Vorgehensweise

Allein die Größe eines ERP-System erschwert eine allumfassende detaillierte Analyse und macht ein manuelles Vorgehen unmöglich. Daher wurden alle Komponenten über die API-Schnittstelle von Forms oder das XML-Format von Reports extrahiert und in einem Repository gespeichert. Dieses wurde über einen simplen Datenbank-Connect zusätzlich mit allen Objekten der Datenbank angereichert. Abschließend hat man die SQL-Aufrufe externer Programme als Textfiles ergänzt. Somit waren in dem Repository alle Objekte der Anwendung mit ihren Eigenschaften und ihrem Code gespeichert, um die Basis einer detaillierten Analyse zu schaffen.

In einem nachgelagerten Arbeitsschritt zerlegt ein Parser den gesamten Sourcecode der Anwendung bis hin zu einzelnen