Infrastruktur-Blueprints für Oracle-Datenbanken

Daniel Steiger
Trivadis AG
Zürich (Schweiz)

Schlüsselworte:

Blueprint, Architektur, Oracle RDBMS, Datenbankinfrastruktur

Einleitung

Konsolidieren, Virtualisieren oder gleich in die Cloud auslagern? Der Auf- oder Umbau einer Oracle-Datenbankinfrastruktur ist mit wichtigen Architekturfragen verknüpft, denen wir in diesem Vortrag nachgehen. Ausgehend von den zentralen, architekturbestimmenden Themen wie Verfügbarkeit, Skalierung, Konsolidierung, Provisionierung, Sicherheit, Kosten und Lifecyle, zeigen wir auf, wie Infrastrukturlösungen für Oracle-Datenbanken, basierend auf den wichtigsten Blueprints und Referenzarchitekturen, entwickelt werden können. Die Verwendung einer soliden Architekturentwicklungsmethodik stellt dabei sicher, dass die verschiedenen Geschäftsanforderungen ihrer Bedeutung entsprechend in die Architektur einfliessen.

Wozu dient ein Infrastruktur-Blueprint?

Um diese Frage zu beantworten muss zuerst einmal geklärt werden, was ein Infrastruktur-Blueprint ist. Da keine allgemein gültige Begriffsdefinition[1] existiert, und es DEN Blueprint so wohl auch nicht gibt, versuchen wir zu umschreiben, was unter „Blueprint“ im Kontext von IT-Infrastrukturen verstanden werden kann:

- Ein Blueprint ist eine abstrahierte Beschreibung von IT-Infrastruktur-Komponenten oder Gesamtinfrastrukturen.
- Ein Blueprint kann als Referenzarchitektur betrachtet werden.
- Ein Blueprint kann herstellerneutral, oder herstellerspezifisch sein.
- Ein Blueprint dient als Kommunikationsmittel zwischen Business und Technologie.
- Ein Blueprint hilft bei der Planung der Infrastruktur.
- Ein Blueprint dient als Basis für die spätere Detailspezifikation.
- Anhand eines Blueprints können Architekturentscheide beurteilt werden.
- Blueprint ≠ Solution. Ein Blueprint ist keine Lösung, dient aber als Basis dafür.

Diese Liste ist natürlich nicht vollständig, aber sie sollte aufzeigen, was ein Blueprint leisten kann und worin sein Mehrwert besteht. Daraus ist auch ersichtlich, dass ein Blueprint in seiner allgemeinen Form (herstellerneutraler Blueprint) starken Modellcharakter hat. Konkretisiert wird das Modellhafte durch die Umsetzung in herstellerspezifische Blueprints. Dort liegt der Fokus dann natürlich auf dem jeweiligen Technologieportfolio und auf technischen Features, mit denen der Hersteller sich von anderen Herstellern zu unterscheiden versucht. Oracle verfügt mit dem Oracle RDBMS und den

[1] Siehe auch http://en.wikipedia.org/wiki/Blueprint

dazugehörigen Features, Optionen und Zusatzprodukten über ein sehr breites Architekturspektrum, welches wir in den folgenden Abschnitten genauer beleuchten werden.

Methodisches Vorgehen führt zum Ziel

Die größte Gefahr – und Versuchung – bei der Entwicklung von Infrastrukturarchitekturen besteht darin, gleich zu Beginn des Architekturprozesses einen technologiegetriebenen Weg einzuschlagen. D.h. ohne sorgfältige Klärung der Anforderungen und der Einflussfaktoren eine „Lösung" auf Basis bekannter oder verfügbaren Technologien zu entwickeln. Dieser sog. Bottom-Up-Ansatz führt nicht nur zu oftmals unbefriedigenden Ergebnissen, sondern widerspricht auch sämtlichen unternehmerischen und planerischen Grundsätzen. Die Losung heißt daher Top-Down[2]: ausgehend von den Geschäfts- und Applikationsanforderungen und unter Berücksichtigung der organisatorischen und technischen Rahmenbedingungen wird die Infrastruktur Schritt für Schritt entwickelt. Entscheidend ist dabei die enge Abstimmung mit den Stakeholder[3] und das schriftliche Festhalten der Ergebnisse und Entscheidungen.

Abb. 1: Designprozess für Infrastrukturarchitekturen

Der Architekturprozess kann in folgende fünf Phasen gegliedert werden:

- **Geschäftsanforderungen analysieren**
  Erfassen und Verstehen der zentralen, nicht-funktionalen Anforderungen wie Verfügbarkeit, Sicherheit oder Performance. Grundlage dazu bilden typischerweise die Geschäftsanforderungen und die SLAs. Bestehende Vorgaben müssen in dieser Phase hinterfragt und allenfalls korrigiert werden.
- **Systemumfeld und Rahmenbedingungen klären**
  Logische und physische Schnittstellen zu Umsystemen, sowie die organisatorischen und technischen Rahmenbedingungen müssen geklärt und dokumentiert werden. Bestehende Standards, Unternehmensrichtlinien und Applikationsanforderungen werden in dieser Phase erfasst und dokumentiert.
- **Architekturentscheide fällen**
  Aufgrund der ermittelten Anforderungen und Rahmenbedingungen werden grundlegende Architekturentscheide gefällt. Darunter fallen Redundanz, Plattformauswahl, Produkteevaluation und Releases, Jeder Entscheid muss begründet und alternativen Lösungen gegenübergestellt werden.
- **System-Design erstellen**
  Die Systemarchitektur wird nun graphisch und textuell beschrieben. Der Abstraktionslevel (Detailierungsgrad) muss sich dabei an der jeweiligen Zielgruppe (Stakeholder) orientieren. Dies bedeutet, dass die verschiedenen Stakeholder und ihre Informationsbedürfnisse bekannt sein müssen und dass Auswahl, Form und Inhalt der Architekturdokumentation entsprechend variiert werden muss.

[2] Das „Top-Down"-Vorgehen wird auch in den bewährten Enterprise Architecture Frameworks wie Zachmann und TOGAF propagiert. Zentrales Element ist dabei das Anforderungsmanagement.
[3] Die Stakeholder sind jene Personen oder Organisationseinheiten, die von der Architektur betroffen sind, oder zu deren Gestaltung beitragen.

- **Implementierung überwachen und bewerten**
  Die Umsetzung der Architekturvorgaben durch Systemingenieure, Fachspezialisten und Testern bedingt einen ständigen Austausch (und auch Überwachung) durch den Infrastrukturarchitekten. Die aus diesem „Reality-Check“ gewonnenen Erkenntnisse müssen wiederum in den iterativen Designprozess einfliessen. Nicht zu vernachlässigen ist zudem die frühzeitige Abstimmung mit den Migrations- und Betriebsverantwortlichen.

Der aufgezeigte Prozess ist hochgradig iterativ. Nach jedem Schritt müssen die Ergebnisse mit den jeweiligen Stakeholder abgestimmt werden, neue Anforderungen oder Rahmenbedingungen müssen beurteilt werden und das Systemdesign muss ggf. angepasst werden. Mehrere Iterationen sind beim Entwurf von komplexen Systemen durchaus normal, gewollt und teilweise auch erforderlich.

Architekturbestimmende Themen

Bei den architekturbestimmenden Themen stellt sich die Frage, welche Geschäfts- oder Applikations-Anforderung die künftige Systemarchitektur in welchem Mass bestimmt. Um die Vielfalt der Anforderungen auf eine sinnvolle und übersichtliche Anzahl zu reduzieren, haben wir aufgrund unserer Erfahrung aus Infrastrukturprojekten, die folgenden architekturrelevanten Themenbereiche identifiziert:

- **Verfügbarkeit**
  Die Verfügbarkeit zählt zu den wichtigsten architekturtreibenden Faktoren. Auf Basis der erforderlichen Serviceverfügbarkeit (siehe folgende Tabelle) und den

Availability %	Downtime per year	Downtime per month*	Downtime per week
98%	7.30 days	14.4 hours	3.36 hours
99%	3.65 days	7.20 hours	1.68 hours
99.5%	1.83 days	3.60 hours	50.4 min
99.9%	8.76 hours	43.2 min	10.1 min
99.99%	52.6 min	4.32 min	1.01 min
99.999%	5.26 min	25.9 s	6.05 s
99.9999%	31.5 s	2.59 s	0.605 s

  Business Continuity Anforderungen, lassen sich wichtige Infrastrukturentscheide ableiten. Die Verfügbarkeitsanforderungen werden typischerweise in einem SLA genau festgelegt. Dabei sollte auch die maximale Ausfallzeit pro Incident (ungeplant), die geplanten Wartungsfenster, und die maximale Anzahlt Incidents pro Zeitspanne spezifiziert werden. Im Falle eines Disasters (z.B. Komplettausfall eines Rechenzentrums), werden die beiden Metriken RPO (Recovery Point Objective) und RTO (Recovery Time Objective) verwendet. RPO definiert den maximal zulässigen Datenverlust; RTO definiert die maximale Serviceausfallzeit[4]. Aufgrund der RTO-Anforderung können mit Hilfe der BC-Tier-Klassifizierung (siehe Abbildung 2) Hinweise auf die einzusetzende Technologie gewonnen werden.

[4] Siehe auch BC Tier 1-7 unter http://en.wikipedia.org/wiki/Seven_tiers_of_disaster_recovery

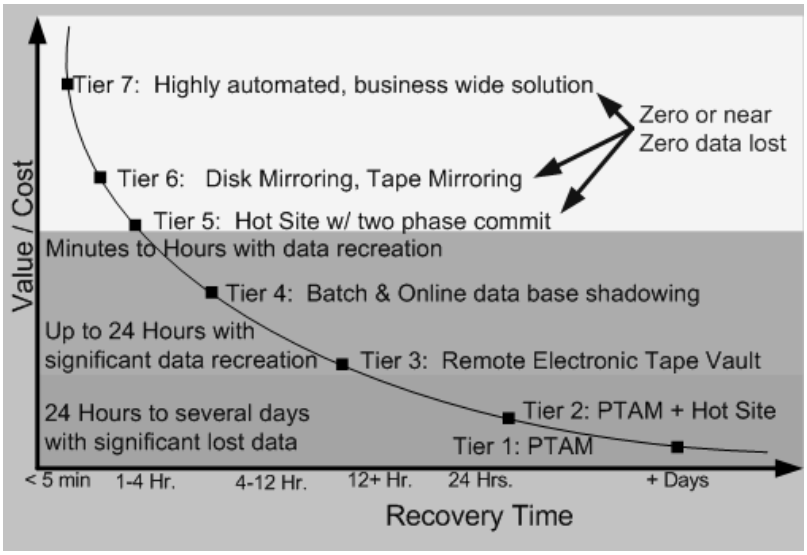

Abb. 2: The 7-Tiers of Disaster Recovery (Quelle: recoveryspecialties.com)

- **Skalierung**
  Die Skalierbarkeit eines Systems äussert sich in der Eigenschaft, die Kapazitäten (CPU, I/O, Memory) bei Bedarf flexibel zu erhöhen, oder zu verringern. Das Ziel ist die Optimierung der Auslastung und der Kosten. Grundlage dazu liefern Kennzahlen aus dem Capacity-Management und -Planning. Als Skalierungsstrategie bieten sich grundsätzlich die vertikale Skalierung (Scale-Up) und die horizontale Skalierung (Scale-Out) an.
- **Konsolidierung**
  Mit dem Zusammenfassen von Datenbanken (oder Datenbankservern) auf gemeinsamer Hardware, soll eine effizientere Nutzung der Ressourcen und der Lizenzen, bei gleichzeitig geringerem Administrations- und Unterhaltsaufwand, erzielt werden. Eine vorgängige Kapazitätsanalyse (CPU, I/O, Memory und Speicher) und ein klares Konsolidierungsziel sind die Voraussetzung für eine erfolgreiche und nachhaltige Konsolidierung.
- **Provisionierung**
  Das schnelle und effiziente Bereitstellen von Datenbankinfrastrukturen wird immer wichtiger. Generell setzen provisionierungsfreundliche Infrastrukturen einen hohen Standardisierungsgrad voraus. Virtualisierung (z.B. mit OracleVM) und die Architektureigenschaften, welche mit Cloud Computing zusammengefasst werden, adressieren diese Nachfrage.
- **Sicherheit**
  Sicherheitsrelevante Anforderungen können die Infrastruktur massgeblich beeinflussen (z.B. physische Trennung von Netzwerksegmenten). Der Einsatz von Sicherheitsfeatures innerhalb der Datenbank (Verschlüsselung, Auditing, Userverwaltung, etc.) ist typischerweise nicht infrastrukturrelevant. Umsysteme, wie beispielsweise PKI-Server, betrachten wir im Zusammenhang mit der eigentlichen Datenbankinfrastruktur nicht.
- **Kosten**
  Kosten sind - mit wenigen Ausnahmen - immer architekturbestimmend und daher architekturrelevant. Neben den Initialkosten müssen auch die wiederkehrenden Kosten für Lizenzen, Wartung und Betrieb berücksichtigt werden. Das Oracle-Lizenmodell ist bezüglich der Kosten ein wesentlicher Einflussfaktor. Die sorgfältige Abklärung der Anforderungen und

Auswahl der eingesetzten Edition und Features ist essentiell. Es gilt sowohl Über- wie auch Unterlizenzierung zu vermeiden. Die Infrastruktur, im Speziellen die Art und die Anzahl der Prozessoren ist aufgrund des Oracle-Lizenzmodells direkt kostenwirksam.

- **Lifecyle**
  Der Lebenszyklus der Applikation muss bei Infrastrukturentscheiden ebenso berücksichtigt werden, wie die Supportzyklen der eingesetzten Produkte (z.B. Oracle Lifetime Support Policy). Ein strategisches Lifecycle-Management auf Unternehmensbasis hilft bei der Klärung der Anforderungen.

Welches architekturtreibende Thema im konkreten Fall wie stark gewichtet wird, muss im Rahmen der Anforderungsanalyse mit den Stakeholdern definiert werden. Wir empfehlen jedes Kriterium zu gewichten (z.B. unwichtig, wichtig, sehr wichtig) und bei der Architekturfindung entsprechend zu berücksichtigen.

Welche Architektur für welche Anforderung?

im Wesentlichen verfügt Oracle über vier Basisarchitekturen: Single-Instanz-Datenbank auf Single-Server, Single-Instanz-Datenbank auf einem Failover-Cluster, Multi-Instanz-Datenbank auf RAC und Standby-Datenbank (Data Guard, für Single- oder Multi-Instanz-Datenbanken). RAC und Data Guard können zudem zur sog. Maximum Availability Architektur (MAA) kombiniert werden. Eine konzeptionelle Erweiterung stellt Oracle VM dar. Oracle VM erlaubt die Virtualisierung von Datenbankservern. In der folgenden Tabelle sind die Charakteristiken der erwähnten Architekturen beschrieben.

Architektur	Charakteristik
Single Instance	Keine Redundanz. Systemausfall auch beim Ausfall einer Komponente. Scale-Up begrenzt möglich (Einsatz von Instance Caging möglich ab 11gR2).
Failover Cluster (FOC)	Knotenredundanz. Service-Failover bei Knotenausfall. Automatischer Failover mit 3rd-Party Clustermanager, mit Oracle Clusterware,oder mit Oracle Failsafe auf Windows. Kein Disasterschutz
Real Application Cluster (RAC)	Skalierungslösung (Scale-Out) und Service-Hochverfügbarkeit. Kein Disasterschutz (ausser mit Streched-Cluster und Host-Based-Mirroring).
Data Guard (DG)	Schutz vor Disaster auf Basis einer aktiven Standby-Umgebung (Standby-DB). Automatischer Failover bei Serviceausfall auf Primärseite (mittels Fast-Start-Failover und Observer)
MAA	Maximum Availability Architecture: Kombination vom RAC und Data Guard. Skalierfähige Daten- und Service-Hochverfügbarkeitslösung
Oracle VM	DB-Server-Virtualisierung auf Basis XEN (Paravirtualisierung für x86-Systeme). Einsatzgebiete: Infrastrukturkonsolidierung, DB-Server-Provisionierungslösung und HA.

Tab. 1: Oracle Basisarchitekturen

In den neueren Releases sind weitere infrastrukturrelevante Features und Softwarekomponenten dazugekommen:

- Grid Infrastruktur (Clusterware und ASM): Mit der Grid Infrastruktur eröffnen sich im Bereich Skalierung, Konsolidierung und Failover sowohl für RAC, wie auch für Failover-Cluster interessante Möglichkeiten (z.B. RAC One Node: Failover mit OMotion erlaubt flexibles Capacitymanagement, oder CRS only: Failoverlösung, bedingt lediglich die Standard Edition, ohne RAC)
- Database Instance Caging: ermöglichst die Limitierung des CPU-Verbrauchs auf Instanzebene. Verfügbar ab 11gR2 (nur EE). Einsatzgebiet: Serverkonsolidierung.

Die Wahl der erwähnten Basisarchitekturen erfolgt in vielen Fällen aufgrund der Verfügbarkeits-anforderungen. Aber auch die Themen Skalierung, Konsolidierung, Provisionierung und Kosten (pro DB inkl. Lizenzen, HW/SW und Betrieb) sind entscheidende Faktoren. Sicherheit und Lifecycle wurden bewusst weggelassen, weil dazu keine allgemeine Aussage gemacht werden kann. Die folgende Tabelle zeigt eine Übersicht über die Eignung (+ bedingt geeignet, ++ geeignet, +++ sehr geeignet) der Basisarchitekturen hinsichtlich der unterschiedlichen Anforderungen:

Architektur	Verfügbarkeit	Konsolidierung	Skalierung	Provisionierung	Kosten
Single Instance	99%	+	+	++	+++
Failover Cluster	99.5%	++	+	+	++
RAC	99.9%	+++	+++	+	+
Data Guard	99.5%	+	+	+	++
MAA	99.9%	+++	+++	+	+
Oracle VM	99.5%	+++	++	+++	+++

Tab. 2: Welche Architektur erfüllt welche Anforderungen?

Building-Blocks - die Bausteine des Blueprints

Building-Blocks sind die Bausteine des Blueprints. Building-Blocks dienen in unserem Fall der (vereinfachten) Darstellung von Datenbankarchitekturen. Der Detaillierungsgrad darf dabei nicht zu klein, aber auch nicht zu gross sein. Die Attribute werden daher – abhängig vom Abstraktionslevel – weggelassen oder aufgeführt. Folgende Grundbausteine[5] werden für die Beschreibung einer Datenbankinfrastruktur benötigt:

Typ	Beschreibung	Attribute
Knoten	physischer oder virtueller Server	DNS-Name, IP-Adresse, Betriebssystem, CPU (Anzahl und Typ), Memory, Typ (physisch oder virtuell)
Service	Datenbankservice	Name
Listener	Datenbanklistener	Name, Port
Instanz	Datenbank- oder ASM-Instanz	SID, Oracle-Home
Clusterware	Oracle Clusterware	Release
OS	Betriebssystem	Vendor, Release, 32/64 Bit
Netzwerk	Privates oder öffentliches LAN oder WAN	IP-Range, Verschlüsselung, Bandbreite
VIP	Virtual IP	IP-Adresse
Datenbank	Datenbankfiles	DB-Name, Oracle-Release, Edition, Typ (RAC, Non-RAC, Primary, Standby)
Storage	physischer oder virtueller Speicher (DAS, SAN oder NAS)	Typ, Kapazität, PIO
Backupsystem	Tapelibrary	IP-Adresse, Typ, Vendor
Location	Datacenter, Rechenzentrum, Gebäude	Bezeichung (Name, Ort, Gebäude), Klassifizierung

Tab. 3: Building-Blocks

Die grafische Darstellung der Building-Blocks wird bewusst einfach gehalten und zeigt nur die zum Verständnis der Architektur notwendigen Komponenten und Details. Das folgende Bild zeigt eine mögliche Darstellung eines physischen Datenbankservers mit einer Datenbankinstanz, Grid-Infrastruktur und SAN-Storage.

[5] Man spricht in diesem Zusammenhang auch von Artefakten.

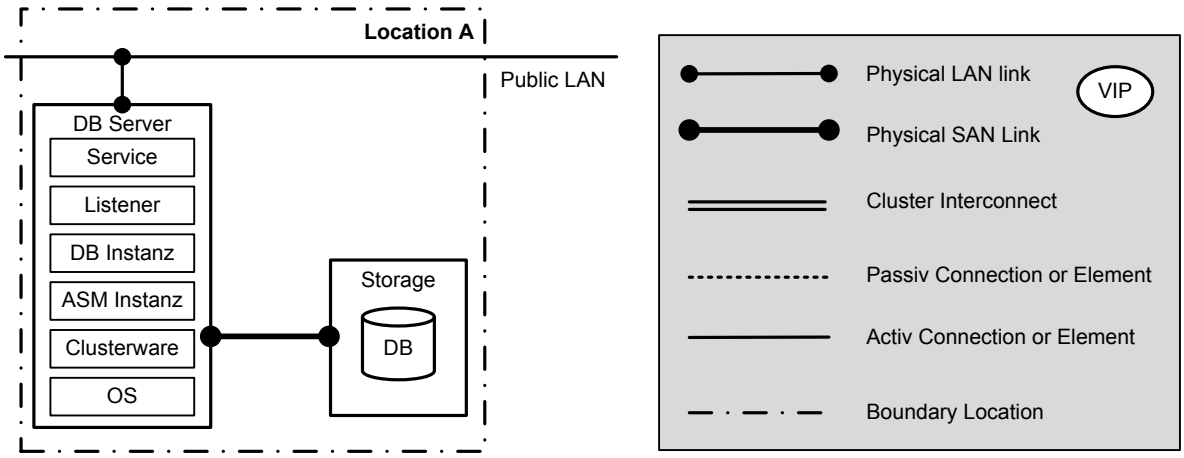

Abb. 3: Darstellung der Building-Blocks

Beispiel: Datenbankserverkonsolidierung

Ausgangslage:
Im Rahmen einer Plattformablösung soll der Ersatz des Datenbankservers geprüft werden. Der Kunde ist offen für den Einsatz neuer Servertechnologien. Zudem wird die Konsolidierung von bestimmten Datenbanken angestrebt.

Bestehende Infrastuktur:
- 2 SUN E6900 mit total 80 Cores Ultrasparc IV+, 320 GB Memory
- Failover-Cluster mit Veritas Cluster und Veritas Filesystem (host based mirroring)
- 31 Datenbanken, 13.1 TB Diskplatz (alloziert), 6.2 TB (benutzt)

Anforderungen:
- RPO: „near zero data loss“
- RTO: 30 Minuten
- Verfügbarkeit: 99.9%
- Skalierbarkeit: Zuwachs der Transaktionsmenge in den nächsten 2 Jahren um den Faktor 3.
- Datenbankkonsolidierung prüfen
- Provisionierungsziel für eine neue Datenbanken: 5 Arbeitstage
- Sicherheit: 2 Datenbanken benötigen strikte Trennung von technischer und applikatorischer Administration
- Kosten: Lizenzoptimierung ist anzustreben
- Nur Einsatz von unterstützter HW und SW

Fragestellung:
- Mit welcher Architektur können diese Anforderungen erfüllt werden?
- Wie sieht ein möglicher Blueprint aus?

Lösung: siehe Präsentation DOAG (Slides)

Fazit und Take-Aways

- Das Mantra im Architekturprozess lautet: Top-Down. Was zählt sind die Geschäftsanforderungen.
- Frameworks wie Zachmann und TOGAF bilden die Leitplanken für den Architekturprozess.
- Reduktion der Komplexität durch Konzentration auf wenige zentrale, architekturbestimmende Themen.
- Blueprints dienen als Brücke zwischen Business und Technologie.
- (Hersteller-)Blueprints sind noch keine Lösung, sondern der Ausgangspunkt dazu.
- Je höher die Anforderungen, desto grösser die Komplexität (und damit die Kosten).

Kontaktadresse:

Daniel Steiger
Senior Consultant
Trivadis AG
Europastrasse 5
CH-8152 Glattbrugg (Schweiz)

Telefon: +41 (44) 808-70-20
Fax: +41 (44) 808-70-21
E-Mail daniel.steiger@trivadis.com
Internet: www.trivadis.com