Rolle Datenbankspezialist
Denis T. Wirries

Rollenbeschreibung und Typisierung eines Arbeitsergebnisses

Inhaltsverzeichnis

Stand: 1. März 2006

Kapitel I Allgemeine Rollenbeschreibung

Benötigte Qualifikationen und Kenntnisse

Es werden gute Kenntnisse über Datenbanksysteme benötigt, sowie deren Verwendung in verschiedenen Einsatzgebieten der Datenhaltung, d.h. man benötigt einen guten Erfahrungsschatz im Datenbankumfeld. Weiter sind Kenntnisse über Integration, Migration und Erweiterung von Datenbanksystemen von großer Bedeutung.
Außerdem sollte man Erfahrung im Bereich der Aufteilung in konzeptionelles bzw. logisches und physisches Datenbankschema mitbringen, damit man in der Entwurfsphase die Designer unterstützen kann.
Als Modellierungssprache sollte man UML, Datenflussdiagramme und Entity-Relationship-Diagramme verwenden und kennen, damit ein Zusammenarbeiten der beteiligten Projektmitglieder reibungslos funktioniert.
Wissen über verschiedene Datenabläufe innerhalb von Standardgeschäftsprozessen könnte sich auch beim Entwurf als außerordentlich nützlich erweisen, da es hier oft Muster für diese Abläuft gibt.

Aufgaben und Anforderungen

Der Datenbankspezialist **unterstützt** bei der Implementierung

- der Datenbankschemata,
- von GP Einschränkungen (Trigger, Constraints),
- Implementierung der Sicherheitsrichtlinien,
- sowie bei der Verbindung zwischen Datenbank- und Anwendungssystem bzw. Neu- und Alt-System und
- der Sicherung der Daten (im Backup-Konzept).

Der Datenbankspezialist leistet während der gesamten Entwicklung des Systems **Hilfestellungen** bei der Entwicklung der Datenbanken und erledigt **datenbankspezifische Aufgaben**.

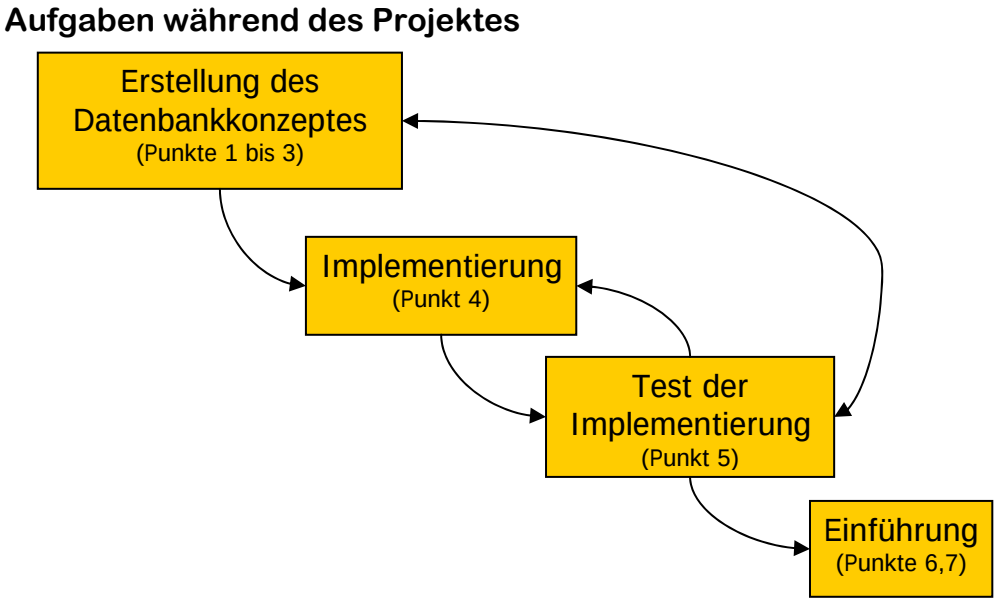

Abbildung 1 – Ablauf im Projekt

1. **Erfassen des Ist-Zustandes** der vorhandenen Datenbanken(-Systeme), Erstellung in Zusammenarbeit mit dem IT-Architekten, dem Integrationsexperten und den Fachbereichsvertretern ein Dokument, das eine **Übersicht über das vorhandene System**

enthält (*Datenbankspezialist nur: → Datenbanken, Server, evt. Parameter, Lasten, usw.*).

2. **Analyse und Grobentwurf** des Datenflusses für die späteren Datenbankenanwendungen unter Berücksichtigung der Geschäftsmodelle, der vorhandenen Systeme (bzw. Datenquellen), der Anforderungen an die Datenbanken und dem IT-Konzept, die der Business-Architekt und IT-Architekt in Abstimmung mit den Fachbereichsvertretern festlegt haben (→ *Anforderung für die Datenbank (z.B. Verteilung, Sicherheit, Lasten, usw.), grobe ER-Diagramme und Datenflusspläne*).
3. **Design der neuen Datenbank (Feinenwurf)**, Erstellung eines Implementierfähigen Konzeptes, das alle im Punkt 2 gefundenen Anforderungen erfüllt (→ *Erstellung des Relationen-Modell bzw. Datenbank-Konzeptes mit Integritätsbedingungen, Datentypen, Funktionen, Sichten, sowie aller weiteren funktionalen bzw. nicht-funktionalen Anforderungen*).
4. **Implementierung** des Designs, Relationen, Constraints, Trigger, benutzerdef. Datentypen, Gespeicherte Prozeduren (→ *implementierte Datenbank*).
5. **Test der Implementierung** (→ *Testvorschriften und Testprotokoll*)
   - Test auf interne Funktionalität, sind alle Funktionen des Designs vollständig und fehlerfrei umgesetzt worden.
   - Test auf Funktionalität mit der Anwendung, Stresstest, Verhalten der Datenbank unter hohen Lastsituationen, in Abstimmung mit dem Test-Manager.
6. **Sicherheitsrichtlinien festlegen**, die vom Sicherheitsbeauftragten vorgeben werden. (→ *Sicherheitsvorschriften und Testprotokoll*)
   - Test der Sicherheit, mit dem Test-Manager abgestimmt.
7. **Erstellen des RollOut-Paketes**, sowie Planung des Einführung und in Betriebnahme der Datenbank und ihrer Funktionalitäten, in Abstimmung mit dem Projektmanager (→*RollOut-Paket der fertigen Datenbank*).

Qualitätssicherung der Aufgaben

1. Gut gemacht, wenn das vorhandene System vollständig untersucht und dokumentiert wurde.
   Testmethode: Inspektion oder Review, Schreibtisch-Test mehrer unabhängiger Gutachter, Walkthrough mit dem Kunden.
2. Gut gemacht, wenn bei der Analyse alle Anforderungen an die Datenbank herausgefunden wurden und Richtig in den Grobentwurf eingeflossen sind.
   Testmethode: Walkthrough mit dem Kunden, Experten-Review.
3. Gut gemacht, wenn alle Anforderungen im Feinentwurf gut umgesetzt wurden.
   Testmethode: Experten-Review.
4. Gut gemacht, wenn alle Anforderung in der Implementierung umgesetzt wurden.
   Testmethode: Automatisierte Testfälle (soweit möglich, sonst manuell), Review oder Inspektion, evtl. auch Walkthrough mit dem Kunden.
5. Gut gemacht, wenn alle Fehler gefunden wurden und dokumentiert sind.
   Testmethode: Automatisierte Testfälle (soweit möglich, sonst manuell), evtl. auch Walkthrough mit dem Kunden.
6. Wie Punkt 5.
7. Gut gemacht, wenn die Einführung und die Betriebsaufnahme reibungslos und ohne Probleme funktionieren, bzw. alle möglichen auftretenden Probleme erkannt und berücksichtigt sind.
   Testmethode: Experten-Review der Planung.

Der wichtigste Projektprozess

Ist der Designprozess, d.h. die Modellierung der Anforderungen an die Datenbank in einem Konzept (Punkt 3), der zu implementierenden Datenbank. Zu einem dient dieses Konzept als

Schnittstelle zwischen den Datenbankprogrammieren und den Anwendungsprogrammieren, also wichtigste Voraussetzung für Implementierung des Systems. Zu anderen wird es als Validerungsgrundlage für die späteren Entwicklungsschritte verwendet (wie z.B. Implementierung und Tests).
Der Prozess soll eventuelle fehlerhafte oder fehlende Anforderungen an die Datenbank finden und diese mit dem Kunden beheben und abstimmen, damit evt. spätere teuere Änderungen vermieden werden können. Der Prozess soll allgemein die groben herausgearbeiteten Anforderungen an das System untersuchen und die Teile, die mit der Datenbank zutun haben, dann in ein geordnetes Konzept übersetzen, an das sich alle halten müssen.
Der Prozess verlangt ständige Abstimmung und Kommunikation mit den anderen Projektmitgliedern und dem Kunden.

Faktoren für den Erfolg der Rolle

- Gewissenhafte Erfassung, Analyse und Umsetzung der Anforderungen in den Datenbankentwurf, da dies die Validierungsgrundlage für alle späteren Schritte ist. 💣
- Absprache mit den anderen Projektrollen um Missverständnisse frühzeitig zuerkennen und zu beseitigen.
- Ständige Überprüfung der umgesetzten Anforderungen in der Datenbank auf Vollständigkeit, Fehlerfreiheit.
- Gewissenhafte Tests der Funktionalität, Sicherheit und des Verhaltens unter Last.

Projektspezifische Aufgaben

- Optimierung von Datenbank z.B. in Hinblick auf Performance beim Umgang mit sehr großen Datenmengen.
- Erstellung und Wartung von Indizes, Analyse und Optimierung von Abfragen und des Sperrverhaltens.
- Erstellung und Wartung komplexer Scripts.

Projektübergeordnete Aufgaben

- Ständige Marktbeobachtung und Evaluierung neuer Versionen, Patches, Support für die Entwicklung
- Durchführung von Schulungen für andere Entwickler
- Consulting in Datenbankfragen, z.B. für die logische und physische Datenmodellierung

Kapitel II Umgebungen

Wichtige Umgebungen für die Rolle:
Physische Umgebung: Netzwerk, weniger wichtig während der Entwicklung, nur wenn verteilte Anwendung / Datenbank Bezug auf diese nehmen, dann aber sehr wichtig.

Virtuelle Entwicklungsumgebungen: Sehr wichtig, denn in dieser Umgebung (Betriebsysteme, Datenbanksysteme, Connectoren, Testumgebung (mögliche Umgebungen beim Kunden)) wird ja entwickelt und getestet.

Organisatorische Entwicklungsumgebungen: Sind weniger wichtig, nur für eventuelle Abläufe (Workflows), die abgebildet werden sollen (nicht unbedingt Gegenstand der Entwicklung dieser Rolle), und Einschränkungen, z.B. Sicherheitsberechtigungen (auch nicht Gegenstand der Entwicklung dieser Rolle).

Methodische Entwicklungsumgebungen: Sind wichtig für das Design (UML, ER-Diagramme) (Analytische Vorgehensweisen), das man eine „gemeinsame Sprache“ spricht, in das Design ist, sowie für die Testmethoden, die zusammen mit dem Test-Manager festgelegt werden, damit die Test für alle nachvollziehbar sind und durch Heuristiken eine „fast“ vollständige Testabdeckung erreicht werden kann.

Kognitive Entwicklungsumgebungen: Natürlich sollte man als Datenbank-Spezialist eine gewisse Erfahrung mitbringen, damit man nicht jede Kleinigkeit nachlesen muss und gewisse Zusammenhänge leichter versteht.

Emotionale Entwicklungsumgebungen: Nur unter den Projektmitgliedern, nicht unbedingt Projekt bezogen.

7 Erfolgsfaktoren für die Gestaltung dieser Umgebung

1. Man sollte sich mit Datenbanken auskennen und Erfahrungen haben (*Kognitive*)
2. Modellierungssprache sollte beherrscht werden (*Methodische*)
3. Testmethoden und –Heuristiken sollten klar sein (*Methodische*)
4. Verständnis über betriebliche Abläufe (Workflows) haben (*Organisatorische*)
5. Verständnis über sicherheitskritische Daten (*Organisatorische*)
6. Kenntnisse von Netzwerkstrukturen (verteilte Anwendungen / Datenbanken) (*Physische*)
7. Richtige (aktuelle und funktionierende) Versionen und Konfigurationen des Betriebssystems, der Datenbank und den Entwicklungstools, sowie den Testumgebungen (*Virtuelle*) (**Wichtigste Voraussetzungen, ohne die nichts geht!**)

Kapitel III Typisierung eines Arbeitsergebnisses

Replikationsplan

Der *Replikationsplan* setzt sich aus mehreren Teilergebnissen, die zum Teil schon bei anderen Arbeitsergebnissen erstellt sind, zusammen.
Als Grundvoraussetzungen für die Erstellung der *Replikationsstrategie*, die der wesentliche Teil des Replikationsplans ist, dient ein Dokument, das alle Informationen und Randbedingungen des vorhandenen und neuen Datenbanksystems, sowie deren Zusammenhänge zusammenträgt. Dieses Dokument nenne ich *Datenbank(-system)-Studie*, da es eine fundierte Analyse des Systems ist, auf derer dann alle weiteren Entscheidungen beruhen. Zusätzlich zur Datenbank-Studie, wird ein Dokument erstellt, das eine Analyse der vorhandenen Anwendungskomponenten des vorhandenen und neuen Systems beinhaltet, dieses nenne ich *Anwendungs(-system)-Studie*.
Zu diesen Dokumenten wird dann ein weiteres Dokument erstellt, das alle Anforderungen an die Replikation festlegt und die Grenzen der Replikation bestimmt, diese Anforderungen folgen aus den Anforderungen des Systems / Projektes, entweder aus funktionalen oder nichtfunktionalen Anforderungen. Dies Dokument nenne ich *Anforderungsdokument der Replikation.*
Aus diesen drei Dokumenten wird nun die Replikationsstrategie erstellt, die Grundvoraussetzung des Replikationsplans. Sie regelt und legt die Replikation der Datenbanken fest. Es werden also alle Anforderungen mit der gegeben Anwendungs- und Datenbanksituation verknüpft und eine Lösungskonzeption, die *Replikationsstrategie*, ermittelt.

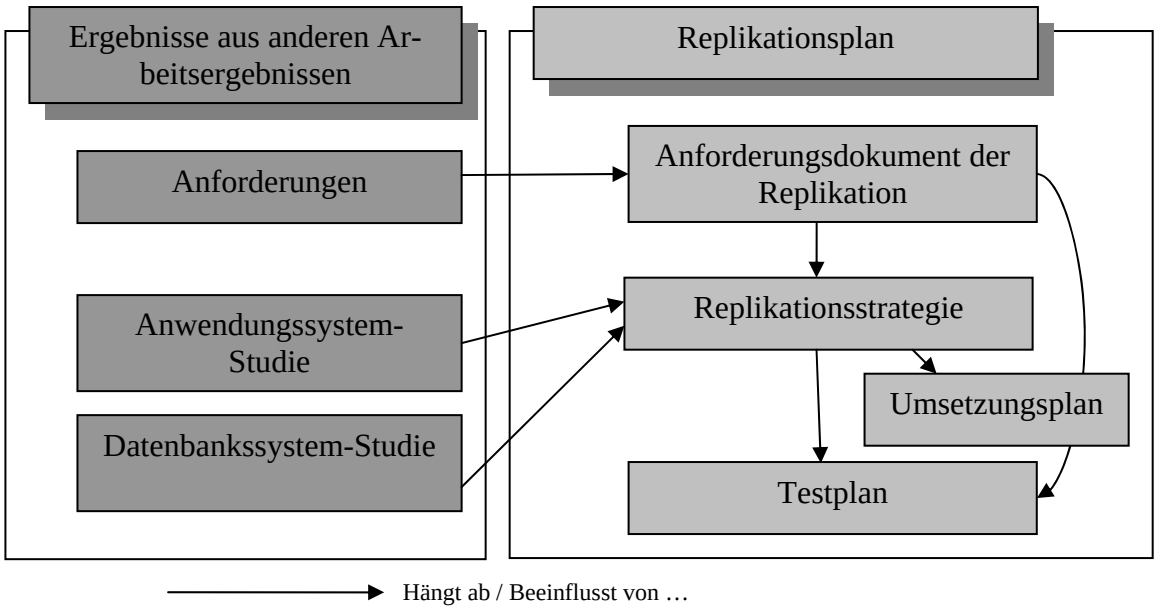

Abbildung 2 - Abhängigkeiten des Replikationsplans

Zum Replikationsplan gehört nun noch ein Dokument, das die Umsetzung und Implementierung der gefunden Replikationsstrategie unterstützt, der so genannte *Umsetzungs- und Implementierungsplan*, sowie ein Testplan, der die Testfälle für die Replikation festlegt.

Eigenschaften des Arbeitsergebnisses

Das Arbeitsergebnis **Replikationsplan** hat folgende Eigenschaften:

- Bestimmung und Festlegung der *Anforderungen an die Replikation*
- Festlegung der *Grenzen der Replikation*
- *Migration* von Datenbanken[1]
- *Anwendungsabhängigkeiten*
- *Klassifizierung* von Datenbanken und Systemen
- *Replikationseigenschaften* (-methoden)
- *Sicherheitseigenschaften*[2]
- Umsetzungs- und Implementierungs*abhängigkeitenplan*
- *Testeigenschaften* der Replikation

Konkrete Ausprägung des Arbeitsergebnisses

Die hier verwendeten Grundvoraussetzungsdokumente enthalten sehr viele verschiedene Informationen, in der Projektwelt werden diese beiden Dokumente wahrscheinlich in viele kleiner strukturierte Dokumente unterteilt, dann werden die benötigten Informationen halt aus diesen entnommen.

Grundvoraussetzung – 1. Ausgangssituation

Das vorhandene und neue Datenbanksystem wurde vollständig erfasst und dokumentiert. Dabei wurde auch speziell auf die Abhängigkeiten zwischen den Systemen geachtet. Als relevante Informationen[3] werden erfasst:

- *Topologie* der Datenbanksysteme (Netzsituation[4] und verteile Systeme)

[1] Meiner Meinung nach, nicht Teil der Replikation, bereits im schon Vorfeld festgelegt.
[2] Nicht unbedient Teil der Replikation, kann hier aber gut umgesetzt werden.
[3] Es werden nur direkte und indirekte benötigte Informationen für das Projekt Call-Center erfasst, alles andere wäre viel zu großer Overhead und würde den Projektumfang sprengen und ist schließlich auch nicht die Aufgabenstellung des Projektes.
[4] Die Verbindungen untereinander, wie Ethernet, SDH, Satellit,…, und deren Auslastung.

- Vorhandene und neue Datenbanken, besonders deren Abhängigkeiten untereinander, aber auch Mapping-Informationen, bei Migration
- Allgemeine Informationen von den Datenbankservern:
  - Datenbanksoftware und –Version
  - Größe der Datenbanken
  - (Volltext-)Indexe, Funktionen, benutzerdefinierte Datentypen
  - Geschäftsregeln, z.B. in Form von Triggern
  - Kataloge und Meta-Daten
- Weitere Informationen folgen aus den Organisatorischen Strukturen:
  - Wartungs- und Batch-Zeiten
  - Relevante Datenbanken für das Call-Center[5]
- Als nächsten werden noch Informationen über die Anwendungen (und –Systeme) benötigt:
  - Anwendungen allgemein und logische Abhängigkeiten
  - Zugriffsmethoden auf die Datenbanken
  - Benötigte Datenbanken
  - Zugriffshäufigkeit
  - Lokalisierung der Anwendungen[6]

Grundvoraussetzung – 2. Anforderungen

Dies ist selbst verständlich die Grundvoraussetzung für das Arbeitsergebnis Replikationsplan, wenn keine Anforderung, funktional oder nicht-funktional, für eine Replikation besteht, wird auch kein Replikationsplan benötigt. Die Anforderungen des Kunden regeln und bestimmen maßgeblich die Replikation, da sie festlegen, wie das Anwendungssystem Call-Center funktionieren soll, bzw. was garantiert werden muss. Der Rest folgt aus den Anforderungen des Anwendungssystems (*indirekte Anforderungen* an die Replikation).
Beispiel: Wenn gefordert wird, das das Call-Center eine hohe Verfügbarkeit und einen schnellen Zugriff auf die Bücherdatenbank haben soll[7], dann muss eine Replikation dieser Datenbank auf den Standort des Call-Centers erfolgen. Generell wäre die Umsetzung der Replikation jetzt noch relativ frei, man sucht nun in den Anforderungen nach weiteren passenden Anforderungen, um die Umsetzung zu verfeinern. Findet man da eine Anforderung, z.B. aus dem Bereich Sicherheit, das Call-Center soll nur auf Informationen von den Büchern, wie Titel und Beschreibung, und nur aktive Einträge zugreifen können. Eventuell findet man auch noch Anforderungen an die Wartung bzw. Aktualisierung der Replikation. Findet man solche Informationen nicht in den Anforderungen des Kunden, dann müssen diese Unklarheiten mit ihm abgestimmt oder aus den indirekten Anforderungen der Anwendung ableiten werden.
Diese so gefundenen Anforderungen werden im nächsten Schritt präzisiert. Wichtig ist auch noch zusagen, das die Replikation oft nicht direkt als Anforderung existiert, sondern eine Folge aus einer oder mehrer Anforderungen ist.

Anforderungsdokument der Replikation

In diesem Dokument werden die oben gefundenen Anforderungen konsistent zusammengefasst, d.h. eventuelle Konflikte zwischen indirekten Anforderungen für die Replikation werden aufgelöst. Dies erfolgt entweder durch prüfen der Anforderungen der Anwendung, also was benötigt die Anwendung, oder in Abstimmung mit dem Kunden, was will der Kunde.

[5] Diese Informationen folgen zum einen aus der Struktur den Datenbanken (indirekt) und zum anderen aus den Anforderungen für das Call-Center (direkt).
[6] Standort der Anwendung, Hauptsitz oder Call-Center
[7] Der ersten Fall, hohe Verfügbarkeit, eine funktionale Anforderung, die messbar ist, macht nicht unbedingt durch eine Replikation notwendig. Nimmt man nun aber noch die zweite Anforderung, schneller Zugriff, funktional oder nicht-funktional, macht eine Replikation Sinn.

Nun hat man eine vollständige Liste aller Anforderungen, die für die Replikation notwendig sind.
Eine Qualitätssicherung dieses Dokumentes lasst sich zum einen auch ein mechanisches Abhaken aller Anforderungen[8] und zum anderen durch ein Experten-Review[9] durchführen.
Beispiel: Von oben, nun konsistent und mit dem Kunden abgestimmt: Die Bücherdaten sollen auf dem Call-Center immer verfügbar und schnell zugreifbar sein, weiter sollen aus Sicherheitsgründen nur Titel und Beschreibung des Buches, sowie nur aktive Einträge, repliziert werden. Die Replikation soll täglich 3x ausgeführt [10] und die Daten können nur auf dem B-Online Standort geändert werden.

Replikationsstrategie

Da man nun alle Grundinformationen und vor allem die Anforderungen für die Replikation zusammen hat, kann man anfangen sich Gedanken über die Replikation an sich machen. Man fängt nun an sich Szenarien aufzubauen, die die geforderten Anforderungen erfüllen und zur gegebenen Situation passen(!).
Hier bei sind generell zwei Ebenen der Ansicht zu unterscheiden:

- Die physische, Hard- und Software-, Ebene
  - Passen die Datenbanksysteme zusammen
  - Unterstützte Replikationsmethoden sind vorhanden, Merge- oder Snapshot-Replikation
  - Online- oder Batch-Replikation
  - Kapazitäten der Verbindungen
  - Lastsituation der Server
- Die logische, Datenbank-, Ebene
  - Transaktionsgestaltung
  - Redundanz der Daten / Aktualität der Daten
  - Relevante Datensätze (z.B. Projektion aus Sicherheitsgründen)
  - Konfliktlösung zwischen Datensätzen

Beispiel: Um das Beispiel von oben aufzugreifen, hier wäre eine Replikationsstrategie angebracht, die die Bücherdatenbank zu einen die nicht benötigten Attribute aus blendet und die Datensätze 3x am Tag abgleicht (Snapshot-Replikation, Push-Abonnent, es werden nur Änderungen zum Abonnenten geschickt, Abonnent kann keine Änderungen vornehmen).
Eine weitere Überlegung der Replikationsstrategie, sind die so genannten *Fail-Safe*-Szenarien, in den man sich überlegt, was bei der Replikation fehlschlagen kann, und wie die Replikation darauf reagieren soll.
Beispiel: In diesem Beispiel nehmen wir an, das die Verbindung zwischen den Datenbankservern öfter ausfällt. Also falls der Push-Abonnent nicht erreicht werden kann, dann versucht der Replikations-Verleger 3x im Abstand von 15 Minuten den Abonnenten zu erreichen, schlägt dies auch fehl wird einen Administrator benachrichtigt.
Qualitätssicherung ist hier auch ein schwieriges Thema, da jede Replikation so ihre ganz eigenen Eigenheiten hat. Am besten ein Review unter Experten und dann evt. noch ein Walkthrough mit dem Kunden.
Kriterien zur Bewertung der Replikationsstrategie wären:

- Einfachheit der Replikation
- Aufwand der Replikation (Hard- und Software / in den Datenbanken selber / Lasten)
- Kosten der Replikation (Verbindungen, Hard- und Software)
- Wartungsanfälligkeit

[8] Test nur auch Vollständigkeit, wurden alle Anforderungen auch betrachtet, es wird kein Inhalt geprüft.
[9] Meiner Meinung nach können die getroffenen Entscheidungen nur doch ein Experten Team überprüft werden, da der Inhalt für mechanische Tests ungeeignet ist.
[10] Die Verbindung sei für das Beispiel hier nicht von Relevanz. Sie existiert und hat ausreichende Kapazitäten.

Umsetzungs- und Implementierungsplan

Hat man nun diese Vorüberlegungen erfüllt kann, man mit einem Umsetzungs- und Implementierungsplan anfangen. Hier werden alle durchzuführenden Schritte, die zur Replikationsstrategie gehören, in einen logischen Ablaufplan gebracht, der ihre Abhängigkeiten untereinander enthalt.

Beispiel[11]:

1. Erstellen einer Sicht[12] für die Datenrelation der Bücher mit dem benötigten Attributen
2. Installieren des Datenbank-Abonnenten
3. Herstellen der Verbindung der Server
4. Einrichten der Replikation der Datensicht
5. (Test der Replikation)[13]

Auch hier ist die Qualitätssicherung nicht so ganz einfach, wie bei allen Plänen gibt es immer Unsicherheiten der Planung, diese kann man nur durch Projekt-Erfahrung und Experten Meinungen beheben.

Testplan

Der Testplan beinhaltet alle Testfälle für die Replikation. Die Testfälle prüfen, ob die zu replizierten Daten vollständig und ohne Fehler repliziert und ob bei Datenkonflikten die richtigen Entscheidungen getroffen werden. Weiter werden die benötigten Anwendungskomponenten getestet, ob sie mit der replizierten Datenbank funktionieren, und ob sich die Datenbank im Fail-Safe-Fall richtig verhält.

Dieser Testplan muss(!) mit dem Testmanager erstellt und koordiniert werden. Qualitätssicherung wie bei jedem Testplan.

[11] Sehr vereinfacht!
[12] Materialisierte Sicht der Datenbank
[13] Nicht direkt unbedingt im Umsetzungs- und Implementierungsplan, befindet sich im Testplan

Kapitel IV Index