

**Blog: Architektur Richtlinien für SharePoint 2010 Anwendungen Teil 2 (SPC 2009)**

Im 2. Teil der Architektur Richtlinien für SharePoint 2010 aus dem Vortrag von Mike Ammerlaan auf der SharePoint Conference 2009, geht es um Architektur Richtlinien für Listen in SharePoint

**Autor: Reiner Ganser**

Im zweiten Teil dieser kleinen Serie geht es um die Architektur Richtlinien für Listen in SharePoint 2010.

- [Teil 1: Einführung und Website Strukturen](#)
- **Teil 2: List Strukturen**
- [Teil 3: Logik, Building Blocks und Benutzerschnittstelle](#)

Dieser Blog ist angelehnt an den Vortrag "Architecture guidance for building applications in SharePoint 2010" von Mike Ammerlaan auf der SharePoint Conference 2009 in Las Vegas.

Die Ausführungen in diesem Teil beziehen sich fast ausschliesslich auf SharePoint 2010.

**SharePoint Liste als Datenablage**

SharePoint Listen werden sehr oft als Datenablage und somit als Ersatz für eine Datenbank verwendet. In der aktuellen Version von SharePoint (MOSS 2007/WSS 3.) führt dies oft zu Problemen, weil die Benutzer sehr viele Elemente in Listen packen und so beispielsweise das Limit von 2000 Einträgen pro Ansicht überschreiten. In SharePoint 2010 wurden deshalb einige Erweiterungen eingebaut, die auch die Arbeit mit sehr großen Listen ermöglichen:

- Lookup to multiple: Es können jetzt auch andere Felder aus einer Liste als Lookup verwendet werden
- Joins innerhalb von CAML. Unter der Haube benutzt LINQ2SharePoint CAML zur Abfrage.
- Eingeschränktes Löschen: Eine Kunde kann nicht gelöscht werden, wenn es zu ihm noch Bestellungen gibt.
- Kaskadiertes Löschen: z.B. alle Bestellungen eines Kunden werden gelöscht, wenn der Kunde gelöscht wird.
- Store-level Enforcement: Etliche Eigenschaften sind nun in den Datenspeicher gewandert und werden dort abgeprüft. Z.B. Wurden Pflichtfelder bisher nur im GUI abgefangen. Man konnte sie aber beispielsweise über die Webservice Schnittstelle leer lassen. Nun wird dies auf Datenschichtebene abgefangen und kann somit nicht mehr umgangen werden
- Unique fields: Felder die einen eindeutigen Wert haben müssen
- Zusammengesetzte Indizes: Dies ist dann sinnvoll, wenn mehrere Werte innerhalb einer Abfrage verwendet werden.
- <In> Klausel für die Ausführung von Reverse-Lookups: z.B. Innerhalb einer CAML Abfrage sollen alle Bestellungen zu einem Kunden gefunden werden. Dies kann mit der <IN> Klausel angegeben werden.
- Formel basierte Validierung: Diese erfolgt ebenfalls auf der Ebene der Datenspeicherung und lässt nur Daten zu, welche anhand der hinterlegten Formel zugelassen sind

### Dinge, die nach wie vor nicht möglich sind

Aber es gibt nach wie vor Dinge, die man nach wie vor nicht mit Listen ohne weiteres tun kann:

- Aggregationsabfragen: Gib mir die ersten 500 Elemente und bilde die Summe aller Abstimmungen
- DISTINCT Abfragen: z.B. Gib mir alle eindeutigen Freitexteingaben eines Feldes, die der Benutzer eingegeben hat
- Ansichten und Abfragen über mehrere Listen hinweg
- Berechtigungen auf Feldebene
- Transaktionale Aktualisierungen

### Throttling Limits

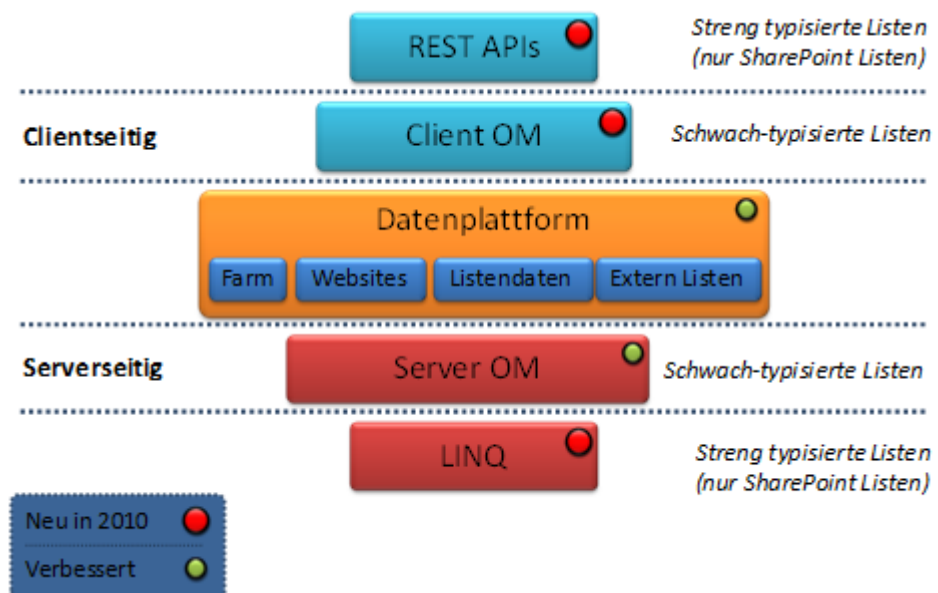
In SharePoint 2010 wurde das sog. Throttling für Listen eingeführt. Es sorgt dafür, dass grosse Listen zwar verarbeitet werden können, jedoch der Server dadurch nicht in die Knie geht. Erreicht wird dies durch Begrenzung von verschiedenen Parametern. Die folgende Aufzählung gibt hierüber einen Überblick:

- 5000 Elemente in einer standardmäßig "Throttled" Abfrage
- 50.000 einzelne Berechtigungen in einer Liste für eine Abfrage
- Bis zu 6 Lookups in standardmäßig "Throttled" Abfrage
- Benutzung von indizierten Spalten oder Ordnern für schnellere Abfragen
- Der Benutzer merkt diese Begrenzungen
- > 100 Elemente auf einer Seite erfordert ein signifikantes Markup Streaming

Sind diese Limits ein Problem für eine Anwendung, sollte die Verwendung von SQL Server in Betracht gezogen werden.

### Daten Zugriffstechnologien

SharePoint 2010 bietet mehr Möglichkeiten, auf die Daten einer Liste, sowohl vom Client, als auch vom Server zuzugreifen. Die folgende Grafik zeigt die einzelnen Technologien und verdeutlicht auch, welche dieser Technologien neu in SharePoint 2010 sind:



Zu beachten in der obigen Grafik ist: LINQ und REST API funktionieren nur gegen SharePoint Listen und nicht gegen externe Listen.

### SharePoint Listen versus Datenbanken

SharePoint Listen sind sehr einfach zu bedienen und verleiten den Benutzer dazu, alle möglichen Daten darin abzulegen. Man muss sich jedoch immer im Hinterkopf behalten, dass die Verwendung von SharePoint Listen einiges an Overhead beansprucht. In vielen Fällen wäre deshalb die Verwendung von Datenbanken deutlich effektiver. Durch die Anbindung über externe Listen ist der Zugriff auch aus SharePoint 2010 möglich. Die folgende Tabelle zeigt eine Gegenüberstellung der beiden Möglichkeiten:

SharePoint Listen	Datenbanken
Dokument Integration (z.B. Funktionalitäten für den Up- und Download von Dokumenten oder Vergeben von Metadaten) Benachrichtigungen bei Änderungen Verwendung von Workflows Berechtigungen können vergeben werden Abruf der Daten über RSS Richer field semantics Easier Setup	Aggregation Distinct Abfragen Transaktionen Bessere Performance Höhere Skalierung Mehr Flexibilität mit Code

### Muster: Daten in Datenbanken speichern und externe Listen in SharePoint benutzen.

- Benutzen Sie Datenbanktabellen mit externen Listen, wenn Sie erweiterte Abfragemöglichkeiten benötigen oder schnelle übergreifende Datenverwendung über Websitesammlungen hinweg.
- Externe Listen können innerhalb von SharePoint angezeigt und durch die Integration von InfoPath auch gepflegt werden. InfoPath eignet sich übrigens auch sehr gut für die Anzeige, vor allem wenn es sich nur um einen Datensatz handelt. Hier kann das FormView Webpart sehr gut eingesetzt werden, welches dann von einer Listenanzeige gesteuert werden kann.
- Es kann ein Webservice mit Business Logic dazwischengeschaltet werden. Dies vereinfacht ggf. den Zugriff und schützt die Daten in der Datenbank.
- **Achtung:** Benachrichtigungen, Workflows und Events werden nicht gefeuert und funktionieren deshalb nicht auf externen Listen

### Muster: Daten in einer grossen Liste innerhalb einer Website speichern

- Wenn gemeinsame Ressourcen über Websitesammlungen hinweg benutzt werden wollen, kann eine versteckte Liste in der Root Website innerhalb einer Websitesammlung verwendet werden. Beispiel: Der Taxonomy Service speichert alle verwalteten Felder innerhalb einer Websitesammlung die Taxonomie Werte in einer grossen Liste.
- Ordner oder indizierte Spalten können verwendet werden, um die Daten besser abfragen zu können

- Lookups oder Reverse Lookups können verwendet werden, um verteilte bzw. verknüpfte Listen zu nutzen
- Überlegen Sie sich auch die Möglichkeit, zu oder von einer Datenbank zu synchronisieren, um das Beste von beiden Welten zu nutzen.

### Tipps und Tricks

- Versuchen Sie Ihre Abfrage genau zu kennen:
  - Was wird in den Webpart abgefragt?
  - Darf der Benutzer seine eigenen Abfragen definieren?
  - Können Webparts Ansichten verwenden
- Diese Punkte können sehr entscheidenden Einfluss auf die Performance haben
- Daraus resultiert auch, welche Felder verwendet werden sollen. "Alle Elemente" oder "Alle Felder für ein Element" sind sehr teuer und sollten vermieden werden.
- Abfragen über mehrere Listen hinweg mit dem SPQuery Datenobjekt sind nach wie vor teuer. Hier ist vor allem auch die Indizierung von den verwendeten Feldern in der Abfrage sehr wichtig.

### Auswahl einer großen SharePoint Liste versus einer Liste in mehreren Websites

- Eine große Liste:
  - Es werden RollUps von Daten über mehrere Websites benötigt
  - Es besteht eine feste Anzahl von Ansichten für die Anzeige der Liste
- Listen in mehreren Websites
  - Es werden individuelle Berechtigungen pro Website benötigt
  - Listen werden ggf. pro Website angepasst (Hinzufügen von neuen Feldern oder Ansichten)
  - Die Synchronisation ist einfacher (wenn eine Website gelöscht wird, wird die Liste auch gelöscht)

### Technik: Synchronisieren zu SharePoint Listen

Dies kann einiges an Arbeit erfordern!!

Es gibt 2 mögliche Wege dies zu tun:

- Ereignisse
  - Direkte Updates
  - Einfacheres Coding
  - Sind jedoch nicht transaktional
- Change Log/GetListItemChanges
  - Erfordert Polling und kann dadurch den Server belasten
  - Besser für den Bau eines von transaktionalen Systemen geeignet

### Ereignisse und Workflows

- Ereignisse
  - Können für die Validierung von Eingaben verwendet werden (Pre-Sync Ereignisse)
  - Können benutzt werden für kurze Aktionen in Post-Ereignissen
  - Weniger Overhead
  - Sollten möglichst wenig Zeit für Ausführung brauchen (möglichst < 1 Min. Ausführungszeit) für Post-Ereignisse
  - Unterstützen Listen- und Website-Aktionen und Änderungen

- Sinnvoll für möglichst einfache und festgelegte synchrone Aufgaben
- Workflows
  - Sind besser geeignet bei lang laufenden Operationen
  - Sind deutlich flexibler und modifizierbar.
  - Bei belastetem System wird der Ablauf der Workflows über Timerjobs gesteuert, so dass eine gewisse Skalierbarkeit bereits eingebaut ist
  - Bieten eine bessere Interaktion mit dem Benutzer (E-Mails, Aufgaben usw.)
  - Standardmäßig nur für Elemente verwendbar. In SharePoint 2010 können Workflows auch für Document Sets verwendet werden und somit für mehrere Dokumente ablaufen.  
Hinweis: Der Start von Workflows auf Ordner Ebene ist sowohl bei MOSS 2007/WSS 3.0, als auch bei SharePoint 2010 programmatisch möglich. In der SharePoint Oberfläche besteht darin standardmäßig jedoch nicht die Möglichkeit.

Diese Ausführungen beenden den 2. Teil dieser Serie. Im [3. Teil](#) geht es um die [Logik, Building Blocks und die Benutzerschnittstelle in SharePoint Anwendungen](#).