

Blog: Architektur Richtlinien für SharePoint 2010 Anwendungen Teil 1 (SPC 2009)

Mike Ammerlaan hat einen interessanten Vortrag auf der SharePoint Conference 2009 mit Hinweisen und Empfehlungen für die Architektur zum Bau von SharePoint Anwendungen/Lösungen gegeben. Dabei sind seine Ausführungen nicht nur für SharePoint 2010, sondern zum großen Teil auch für MOSS 2007/WSS 3.0 relevant.

Autor: Reiner Ganser

Nach einer kleinen Blog-Pause aus Projektgründen, habe ich mich entschlossen, obwohl die SharePoint Conference 2009 vorüber ist, trotzdem noch weiter über die Beiträge zu berichten. Den Anfang macht ein Beitrag von Mike Ammerlaan. Er präsentierte einen interessanten Vortrag mit Hinweisen und Empfehlungen für den Bau von SharePoint Anwendungen/Lösungen. Die Hinweise beziehen sich vornehmlich auf die Entwicklung. Dabei sind seine Ausführungen nicht nur für SharePoint 2010, sondern zum großen Teil auch für MOSS 2007/WSS 3.0 relevant, da die Architektur im Kern bzgl. Websites, Listen usw. mehr oder weniger die gleiche geblieben ist. Dies bedeutet auch für die zukünftige Version von SharePoint, dass man sich im Vorfeld Gedanken über die Architektur (z.B. Der Websitestruktur) machen sollte. Vieles ist zwar in der neuen Version einfacher geworden, grobe Fehlentscheidungen in der Architektur zu Beginn des Projektes verzeiht auch die neue Version nicht und bedingt dann oft großen Aufwand, um das System wieder in die richtige Richtung zu "biegen".

Ich habe mich entschlossen die Ausführungen von Mike Ammerlaan in mehrere Posts zu packen, da die ganze Sache sonst etwas länglich und unübersichtlich geworden wäre. Die Posts sind folgendermassen aufgeteilt:

- Teil 1: Einführung und Website Strukturen
- Teil 2: List Strukturen
- Teil 3: Logik, Building Blocks und Benutzerschnittstelle

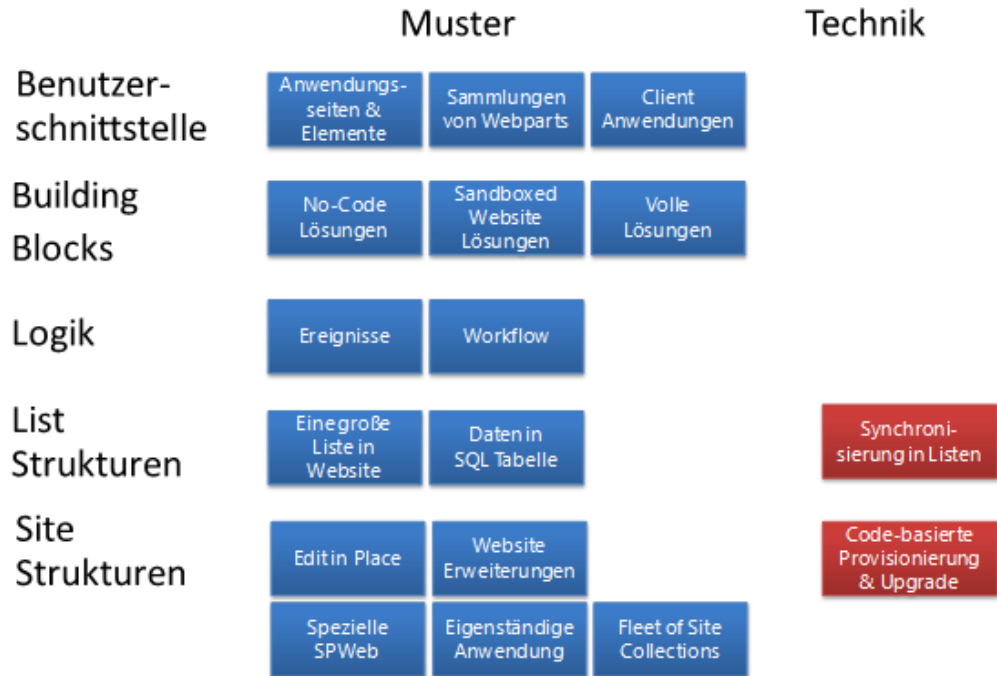
Die folgende Zusammenfassung gibt die Ausführungen von Mike Ammerlaan wieder, erweitert sie stellenweise und gibt gleichzeitig auch Hinweise, an welcher Stelle diese Empfehlungen bzw. Anleitungen auch für MOSS 2007/WSS 3.0 zutreffen.

Die oben aufgeführten Posts behandeln vor allem die folgenden Themen:

- SharePoint Liste oder Datenbank?
- Eine Sitecollection oder mehrere
- No-Code, Sandboxed oder Full-Trust Lösung?

Es gibt dabei keine eindeutige Aussagen. Es ist wie immer: Es kommt darauf an ... Umso wichtiger ist es deshalb, sich zu überlegen, was die Benutzer brauchen und wie sich die Anwendung bzw. Nutzung entwickeln wird.

Die folgende Übersicht zeigt die Bereiche, die in den einzelnen Posts behandelt werden, angefangen von unten nach oben:



Als kleiner Einstieg nochmal die Bereiche, die SharePoint 2010 abdeckt. Dieses Mal allerdings mit etwas mehr an Details:

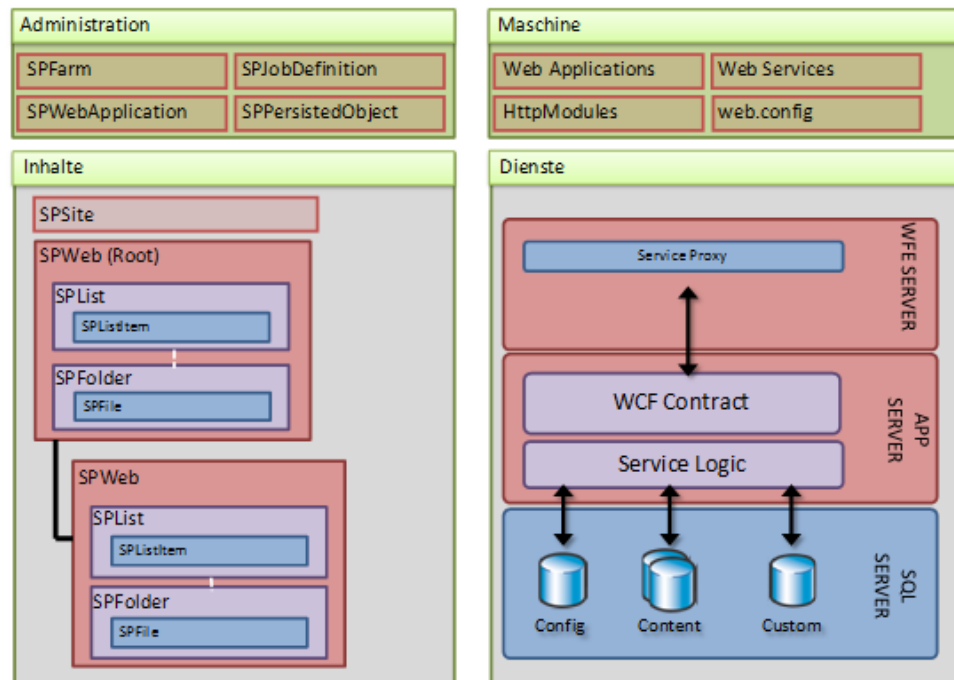


Dies soll einfach nur zeigen, dass SharePoint 2010 aus sehr vielen Komponenten und Technologien besteht und es selten nur einen Weg gibt, der zum Ziel führt. Vielmehr

muss man die Implikationen der einzelnen Lösungswege kennen und sich dann für eine Vorgehensweise entscheiden. Diese Punkte versucht Mike Ammerlans Vortrag aufzugreifen.

Teil 1: Website Struktur

Die folgende Grafik zeigt einen Überblick über die verschiedenen Bereiche, die es für die Strukturierung von SharePoint Websites zu beachten gilt:

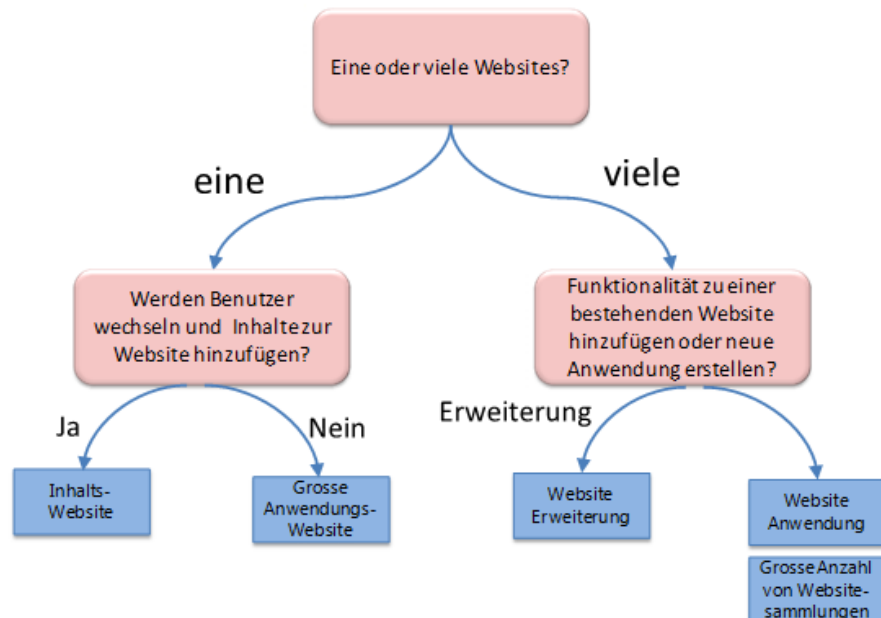


Auf der linken Seite im Bereich Inhalte ist die interne Struktur anhand des Objektmodells aufgezeigt. An oberster Stelle steht das SPSite Objekt, welches in gewissem Masse die Websitesammlung repräsentiert. Die eigentlichen Inhalte befinden sich in SPWeb Objekten, die beliebig tief inenander geschachtelt sein können. Der Bereich Administration bezieht sich auf die zentrale Administration. Als Entwickler kann man hier die gezeigten Objekte nutzen, um Automatisierungen vorzunehmen. In SharePoint 2010 kann die zentrale Administration durch eigene Objekte erweitert werden durch Ableiten von SPPersistentObject. Die rechte Seite im Bereich Maschine zeigt die Möglichkeiten auf Seiten von ASP.NET. Hier integriert sich SharePoint 2010 noch stärker durch Integration in die Pipeline von IIS. Gerade diese Möglichkeit hätte man sich bei MOSS 2007/WSS 3.0 des öfteren gewünscht. Dadurch lassen sich nun erheblich mehr Möglichkeiten auf Protokollebene ausschöpfen, da die HTTP Module mehr Ereignisse kennen. Im Bereich Dienste hat sich in SharePoint 2010 einiges getan: Die neue Service Infrastruktur (siehe auch [SharePoint Conference 2009 Las Vegas Teil 10: Einführung in die Service Applications und Topologie](#)) ermöglicht die Auslagerung von Diensten, die viel an CPU Zeit brauchen, oder eine spezielle Infrastruktur benötigen. Zusätzlich lässt sich die Funktionalität von SharePoint 2010 anhand eigener Dienste erweitern.

- **Service Infrastruktur:** Die neue Service Infrastruktur von SharePoint 2010 bietet deutlich mehr Möglichkeiten Funktionalitäten zu verteilen und Server zu entlasten (siehe auch [SharePoint Conference 2009 Las Vegas Teil 10: Einführung in die Service Applications und Topologie](#)). Die Frage ist, wann sollen Dienste genutzt bzw. angeboten werden:
 - Zugriff auf gemeinsame Daten ausserhalb einer SharePoint Website soll angeboten werden
 - Spezielle Berechnungen oder intensive CPU Nutzung wird benötigt
 - Bei langlaufenden (> 5 Minuten) asynchronen Operationen
 - Scale Out Strategie wird benötigt
- **Typen von Projekten:** Anhand der zu erwartenden Nutzung kann man die Anwendungen klassifizieren:

	Eine Website	Viele Websites
Existiert bereits oder eine mitgelieferte Vorlage wird verwendet	z.B., <i>Internet Website, Portal</i> Inhalts-Website	z.B., <i>Artikelverwaltung in Teamsites</i> Website-Erweiterung
Von Grund auf gebaut	z.B., <i>Mitarbeiter Bewertungs-Website</i> Grosse Anwendungswebsite	z.B., <i>Partner Extranet Websites</i> Website Anwendung Spezialitäten Website Grosse Anzahl von Websitesammlungen

- Anhand der obigen Übersicht lässt sich die folgende Entscheidungsstruktur für die grundlegende Entscheidung der Website Struktur ableiten:



- Die einzelnen Websitearten werden im folgenden noch etwas näher betrachtet:
- **Mustervorgehen: Inhalts-Website (gilt auch für MOSS 2007)**
 - Für eine einzelne große Instanz einer Website ist die Editierung sehr einfach und durchgängig.

- Es sollte vermieden werden, zu viele Artefakte zu verwenden (z.B. Dinge die der Benutzer eingibt, Elemente, die über Solution Packages eingebracht werden, Dinge, die installiert werden müssen, Dinge die konfiguriert werden müssen auf den verschiedenen Ebenen usw.). Zu viele Artefakte erhöht die Komplexität einer Website deutlich, da Inhalte aus der Inhaltsdatenbank und von den Entwicklern erstellte Komponenten synchronisiert werden müssen.
- Für die Migration von Inhalten von der Staging in die Produktionsumgebung sollte der Mechanismus Content Deployment verwendet werden
- Vorgefertigte Elemente (z.B. Webcontrols) sollten über Solution Packages verteilt werden
- See slides
- Anstatt vieles in Webparts zu packen (vor allem, wenn hohe Last erzeugt wird), kann nun die Service Infrastruktur genutzt werden, um diese zu trennen -> Service kann bei hoher Belastung auch auf einen anderen Server gelegt werden
- **Artefakte**
 - Von Autoren erstellte Artefakte (Webseiten)
 - Werden im Browser erstellt
 - Sind abgelegt in der Inhaltsdatenbank der Website
 - Verwaltet anhand von Backup/Restore
 - Werden verteilt anhand von Content Deployment
 - Lösungsartefakte (z.B. Webparts, Webcontrols)
 - Werden von Entwicklern erstellt
 - Oftmals viele im Standard vorhandene Elemente
 - Werden in Source Code Verwaltung verwaltet
 - Werden verteilt/installiert anhand von SharePoint Solutions
 - In einer Inhalts-Website werden immer vom Benutzer erstellte Inhalte existieren. Dies ist ein unterschiedliches Modell zur Software Entwicklung, bei dem die Komponenten anhand von SharePoint Solutions verteilt werden. Inhalte per SharePoint Solutions zu verteilen ist zumeist umständlich, wenn nicht sogar in vielen Fällen unmöglich.
- **Mustervorgehen: Grosse Anwendungs Website (Gilt auch für MOSS 2007/WSS 3.0)**
 - Eine Anwendungswebsite hat eine feste Funktionalität, die dem Benutzer hilft eine spezielle Aufgabe zu erledigen. Beispiel: Datensatzrepository
 - Layouts-Seiten (Seiten auf der Festplatte des Webserver im Verzeichnis /layouts) oder Webparts für die Benutzerschnittstelle
 - Die Anzahl der Designer und Website Administrator sollte minimiert werden. Die Regel sollte sein, dass jeder Benutzer max. Bearbeiter in der Website ist.
 - Sollte in einer eigenen Websitesammlung laufen. Dies ist vor allem aus Performance Gründen zu empfehlen. Nur dann kann z.B. eine eigene Datenbank zugewiesen werden.
 - An dieser Stelle sollte man sich auch immer wieder fragen, ob es sinnvoll ist, die entsprechende Anwendung in SharePoint abzubilden oder ob es nicht besser wäre, eine eigenständige ASP.NET

Anwendung zu erstellen. Die Frage ist an dieser Stelle zumeist: welche Komponenten stellt SharePoint bereit, die für die Anwendung von Nutzen ist. Man muss sich an dieser Stelle immer wieder vor Augen führen, dass SharePoint auf ASP.NET aufbaut. Zudem können Teile von SharePoint auch aus einer ASP.NET Anwendung heraus genutzt werden können (z.B. Dokumentbibliotheken für die Ablage von Dokumenten) , ohne deswegen gleich die gesamte Anwendung auf Basis von SharePoint umzusetzen.

- **Mustervorgehen: Websiteerweiterungen (Gilt auch für MOSS 2007/WSS 3.0)**
 - Sollte dann in Betracht gezogen werden, wenn eine bestehende Funktionalität von SharePoint angepasst oder erweitert werden soll. Beispiel: Die Kennzeichnung einer Website als öffentlich, intern oder vertraulich.
 - Es sollte ein sog. "Stapeled Feature" (Feature, welches aufgerufen wird, wenn ein anderes Feature aktiviert wird) oder eine benutzerdefinierte Website Definitionen verwendet werden, um die Anpassungen in SharePoint einzubringen
 - **SharePoint 2010:** Es gibt neue Ereignisse, die helfen können, eine Funktion umzusetzen. So gibt es nun die Ereignisse WebAdd und ListAdd.
 - Über Delegate Controls, Custom Actions und feature Activation lässt sich selbst erstellter Code einbauen
 - Typischerweise werden Websitesammlungs- und Website Features benötigt, um die Funktionalität umzusetzen.
 - Eine Frage, die man sich hier stellen muss: Ist es notwendig, dass die neue Funktionalität auch von den bereits bestehenden Websites genutzt werden kann? Ist dies der Fall, sollte die Aktivierung der neuen Funktionalität für die bestehenden Websites nicht im Installationsprogramm oder eine Feature receiver passieren. Besser ist es aus meiner Sicht ein spezielles "Progrämmchen" zu erstellen, welches die Funktionalität aktiviert. Oder man baut die Funktionalität in einer zentralen Komponente ein, die von allen Websites genutzt wird (z.B. Ein Web Control in der Masterseite).
- **Mustervorgehen: Websiteanwendung (Gilt auch für MOSS 2007/WSS 3.0)**
 - Dies sollte genutzt werden für spezielle Aufgaben. Beispiele aus SharePoint sind Besprechungsarbeitsbereiche, Zugriff auf Web datenbanken oder die Vorgane der Fantastic 40
 - SharePoint nutzt dieses Konzept sehr oft
 - Es werden Anwendungen mit einbfacher Funktionalität und wenig Auswirkungen auf das Gesamtsystem erstellt
 - Es wird eine grosse Anzahl geklonten Anwendungen erwartet (z.B. Viele Besprechungsarbeitsräume)
 - Eine spezielle Website sollte genutzt werden für die Administration und gemeinsame Nutzung von Ressourcen (z.B. Konfigurationseinstellungen)
- **Mustervorgehen: Spezielle Website (Gilt auch für MOSS 2007/WSS 3.0)**
 - Sollte genutzt werden, um Funktionalitäten innerhalb einer größeren Websitesammlung oder Websitesammlungsübergreifend zu aggregieren. Z.B. Suchcenter, Reporting-Center.

- Ressource Listen und Ansichten sollten in eine einzelne Website gebündelt werden, welche diese wieder für die geamte Websitesammlung zur Verfügung stellt.
- In einer solchen Website sollte es nicht möglich sein, Anpassungen vorzunehmen. Sie sollte nur als Benutzerschnittstelle mit Funktionalität oder zur Administration dienen.
- An dieser Stelle kann man sich auch überlegen, ob man eine zusätzlich Websitesammlung bereitstellt, welche die Ressourcen verfügbar macht. An dieser Stelle besteht jedoch die Problematik , dass der Zugriff über Websitesammlungsgrenzen hinweg in SharePoint immer etwas komplizierter ist, als zwischen Websites innerhalb einer Websitesammlung. SharePoint 2010 bringt an dieser Stelle keine wesentlichen Neuerungen.
- **Mustervorgehen: Grosse Anzahl von Websitesammlungen (Gilt auch für MOSS 2007/WSS 3.0)**
 - Sollte genutzt werden, wenn mit sehr vielen Inhalten (z.B. Dokumenten) gerechnet wird oder wenn unabhängige Einheiten verwaltet werden müssen. Z.B. Persönliche Websites; Extranet Partner Websites
 - Es werden viele Instanzen erwartet
 - Websitesammlungen bieten neinige Vorteile, die Websites nicht haben: Können auf Datenbanken verteilt werden, können ein Quota haben
 - Sicherheit: Es kann Host Header Modus verwendet werden, um eine volle die Separierung zu erreichen. Da jede Websitesammlung in diesem Modus eine eigene Domäne erhält, sind Angriffe anhand Cross Site Scripting deutlich schwieriger
 - **Wichtig:** Die Erzeugung von Websitesammlungen dauert im Normalfall deutlich länger als die Erzeugung einer Website. Dies sollte berücksichtigt werden. Abhilfe schafft hier die verzögerte Erstellung einer Websitesammlung (z.B. Erst wenn ein Benutzer die Website tatsächlich braucht, wird diese angelegt), nur die teilweise Aktivierung von Features oder Anlage von Websitesammlungen auf Vorrat (z.B. Durch einen nächtlichen Prozess, der die Websitesammlungen anlegt).
- **Dinge, die man wissen sollte (Gilt auch für MOSS 2007/WSS 3.0)**
 - Websites (SPWeb) sind das Zetrum des SharePoint Universums.
 - Metadaten einer Website werden geladen und gehalten
 - Erzeugen einer neuen Websites im Objektmodell erfordert Zeit
 - Deshalb sind Operationen über mehrere Websites hinweg ressourcen intensiver, als Operationen innerhalb einer Website
 - Unterschiedliche Websites sind unterschiedlich...
 - Unterschiedliche Websites können z.B. unterschiedliche Sprechend nutzen oder verschiedene Berechtigungen haben. Dies führt oft dann zu Problemen, wenn man in der Ebtwicklungsumgebung von idealen Bedingungen ausgeht, die in der Produktioin dann ganz anders sind (z.B. Der Benutzer von Website A hat keine Berechtigungen auf Website B, benötigt aber Ressourcen aus Website B). Man sollte deshalb die Entwicklungsumgebung möglichst genau der Produktionsumgebung nachbilden.

- Websitesammlungen sind nach wie vor eine Barriere, auch in SharePoint 2010
 - Keine Abfragen über Websitesammlungen hinweg über das Objektmodell
 - Eine Abhilfe kann die Nutzung von Silverlight sein, welche Webdienste zur Abfrage der Dsten nutzt. Solange die Websitesammlungen in der gleichen Domäne sind, funktioniert die Abfrage auch anderer Websitesammlungen.
- **Wahl zwischen Websiteanwendung und vielen Websitesammlungen (Gilt auch für MOSS 2007/WSS 3.0)**
 - Die folgenden Punkt geben nochmal einen Gesamtüberblick über die Fragen, die zur Wahl einer Website Hierarchie innerhalb einer Websitesammlung bzw. zur Aufteilung in mehrere Websitesammlungen führen:
 - Website Hierarchie
 - Inhalte kann zwischen Website gemeinsam genutzt werden (Daten, Inhaltstypen, Seiten, Masterseiten usw.). Inhaltstypen können in SharePoint 2010 auch über Websitesammlungen hinweg genutzt werden.
 - Einfacher
 - Einfacher zu verwalten
 - Viele Websitesammlungen
 - Viele und/oder große Inhalte (100rte von GB)
 - Benötigt maximale Sicherheit und Separierung
 - Viele Benutzer, die unabhängige Kontrolle über Anpassungen benötigen
- **Update muss beachtet werden**
 - Ist eines der schwierigsten Punkte in SharePoint, da es eine sehr große Anzahl von Instanzen geben kann, die die Erweiterung nutzt. In SharePoint 2010 können Features eine Version haben und es kann definiert werden, was bei einem Versionswechsel passieren soll. Dies erfolgt anhand eines sog. Feature Upgrade Event Hook, der dazu genutzt werden kann, Code aufzurufen, wenn das Feature ein Upgrade erfährt.
 - Einige Dinge können im Feature XML definiert werden. In den meisten Fällen wird aber Code notwendig sein, um die entsprechenden Aktionen durchzuführen.
 - Es existiert ein sog. Feature Query API. Dies ermöglicht dem Entwickler abzufragen, welche Version die entsprechenden Features haben. Die iterative Abfrage von allen Websites ist also nicht notwendig.
 - Das deinstallieren kann sehr trickreich sein (z.B. Entfernen einer Websitespalte, wenn das Feature deinstalliert wird. Da andere Inhaltstypen dieses Feld nutzen können, würde dies zu Fehlern führen). **Gilt auch für MOSS 2007/WSS 3.0)**
- **Technik: Code basierte Verteilung/Installation**
 - Code basierte Verteilung/Installation stellt sicher, dass die Elemente, welche das entsprechende Feature benötigt auch vorhanden sind. Hierzu kann sehr gut ein sog. Feature Activation Callout verwendet werden.

- Anstatt von XML in der Feature Definition wird zum Aktivierungszeitpunkt des Features Code ausgeführt. Dies kann dazu genutzt werden, um z.B. Die folgenden Elemente zu erzeugen:
 - Inhaltstypen
 - Felder
 - Websitestruktur
 - Einzelne Listeninstanz mit speziellem Schema
- Wenn das Objekt noch nicht existiert wird es erzeugt, wenn es schon existiert wird es für die neue Erfordernisse angepasst.
- Ermöglicht einen kontrollierteren Upgrade.
- Neu in SharePoint 2010: Inhaltstypen haben nun einen public Konstruktor. Dies ermöglicht die einfache Erzeugung von Inhaltstypen mit Code anstatt einer XML Definition.
- Der Nachteil der code basierten Verteilung/Installation kann eine etwas schlechtere Performance bei der Provisionierung des Features sein, vor allem dann, wenn sehr viele Aktivierung durchgeführt werden.
- **Hinweise zur Sicherheit**
 - Gebrochene Berechtigungsvererbung führt in vielen Anwendungen zu Problemen, da Berechtigungen, die eine Anwendung an anderer Stelle benötigt, plötzlich nicht mehr vorhanden ist. Hierzu existiert in SharePoint 2010 ein Flag, welches eine Liste als Application List deklariert. Dieses war bereits in SharePoint 2003 vorhanden, jedoch in MOSS 2007/WSS 3.0 nicht mehr vorhanden. Es restriktiert die Bearbeitung der Liste auf Designer, lässt aber die Benutzer lesend darauf zugreifen. Zusätzlich existiert nach wie vor die Eigenschaft [AllowEveryoneViewItems](#). Dies ermöglicht ebenfalls den generellen Zugriff auf die Ressourcen in der Liste durch die Benutzer, jedoch die eingeschränkte Bearbeitung der Liste. Dies wird beispielsweise in der Masterpage Galerie verwendet.
 - Berechtigungen auf Elementebene sind sehr "teuer". Es sollte möglichst vermieden werden, dieses Feature zu benutzen. Sinnvoll ist dies nur, wenn es sich um eine geringe Anzahl von Elementen handelt.
 - Man sollte die Grenzen der Anwendung kennen. Beispielsweise führt der Versuch Update in der zentralen Konfigurationdatenbank durchzuführen (anhand des Zentraladministrations Objektmodells), aus einem Webpart auf einer Website, in der Produktion zu einem Fehler, da die Rechte fehlen. Hier muss über alternative Möglichkeiten nachgedacht werden (z.B. Die Verwendung von Timerjobs).
 - Elevation und Impersonation können gefährlich sein, vor allem wenn man nicht genügend darüber nachdenkt.
 - SharePoint 2010 limitiert die Editoren einer Website mit Skript zu arbeiten. Diese Möglichkeit ist optional.
 - Werden HTML Seiten in einer Bibliothek abgelegt, werden diese zum Download angeboten und werden nicht in der Browserseite geöffnet
 - Verschiedene Webparts können nicht geändert werden.
 - Anpassungen an ASPX Seiten benötigen Designer Rechte
- **Allgemeines Fragen**

- Mit unterschiedlichen Berechtigungebenen wird nicht korrekt umgegangen
 - Einige Ressource benötigen "AllowEveryoneViewItems"
 - Einige Ressourcen benötigen Impersonation, um den Zugriff sicherzustellen
- Operationen über mehr als 10 Websites (SPWeb) können sehr lange dauern. Hier sollte über Alternativen nachgedacht werden (z.B. Die Verwendung von Timerjobs).

Dies beendet den ersten Teil dieser Serie. Im nächsten Teil geht es um die Strukturen von Listen. Gerade in diesem Bereich bietet SharePoint 2010 einiges Neues.