Konfigurationsmanagement als Garant für Effizienz

Stephan La Rocca
TEAM GmbH
Paderborn

Schlüsselworte
Design, Deployment, Versionierung, Patchmanagement, Pattern

Einleitung
In dem Vortrag werden Synergie-Effekte über Software-Projekte hinweg aufgezeigt. Wiederkehrende Aufgaben zum Design, Versionierung, Deployment, Patchmanagement können identifiziert und dann mit verschiedenen Vorgaben und Vorgehensweisen über eine Vielzahl von Projekten vereinheitlicht werden.
In einem Softwareprojekt befinden sich die Anforderungen an ein Konfigurationsmanagement auf vielen verschiedenen Ebenen. Das beginnt bei ausgewählten Datei-Templates in einer Versionsverwaltung und endet bei Konfigurationsparametern für die laufende Applikation. Dabei ergeben sich z.B. Aufgaben für einen einheitlichen Zugriff oder einer effizienten Überführung bei System- oder Versionswechsel.

Konfigurationsmanagement
Greifen wir die „Definition“ des Begriffes bei Wikipedia auf (http://de.wikipedia.org/wiki/Konfigurationsmanagement#cite_note-1), so treffen die Annahmen, die im Hintergrund der ersten großen Raumfahrtprojekte in den späten 50er Jahren festgestellt wurden, mit einem erstaunlich hohen Deckungsgrad auch heute auf viele Software-Projekte zu.
ZITATE
Leider verkommt das Konfigurationsmanagemt in der Software-Entwicklung (http://de.wikipedia.org/wiki/Software-Configuration-Management) aus meiner Sicht zu sehr in den Bereich Versionsverwaltung. Sicherlich ein wichtiger Bestandteil, aber dennoch nicht alle Facetten.
Auf dem Paperback des Buches (Konfigurationsmanagement mit Subversion, Maven und Redmine: Grundlagen für Softwarearchitekten und Entwickler http://www.amazon.de/Konfigurationsmanagement-Subversion-Maven-Redmine-Softwarearchitekten/dp/3898645215):
Konfigurationsmanagement (KM) gilt als langweilig und teuer und ist kein Thema, bei dem Softwarearchitekten und Entwickler glänzende Augen bekommen. Daher verzichtet man in vielen Projekten auf die Einführung eines KM-Prozesses. In der Folge müssen die Teams typische KM-Themen wie die Festlegung einer Projektstruktur, die Projektautomatisierung und den Umgang mit parallelen Entwicklungssträngen 'nebenher' erledigen. Dies führt aber fast zwangsläufig zu Verzug in der Planung und zu Qualitätsproblemen.

Betrachten wir im folgenden Vortrag ein etwas breiteres Einsatzgebiet des Konfigurationsmanagement. Entlang des Entwicklungsprojektes von der Bereitstellung der Infrastruktur, über die Unterstützung bei der Software-Entwicklung bis hin zum Einsatz während des Produktsystems.

Infrastruktur

Für die Software-Entwicklung bietet es sich an, die verschiedenen Systeme (Entwicklungsumgebung, Testumgebung, Integrationsumgebung, Wartungsumgebung, etc) zu virtualisieren. Durch die einfache

Möglichkeit, virtuelle Systeme zu clonen, bietet es sich an, eine Art Evolution vorzuhalten. Eine Maschine, nur mit Betriebssystem, darauf aufbauend eine incl. Datenbank, darauf aufbauend eine mit der passenden Entwicklungsumgebung.

So ist sichergestellt, dass sich alle Systeme zunächst auch auf dem gleichen Stand befinden. Ebenfalls kann eine Entwicklung in neue Versionen beteiligter Softwarekomponenten leicht erfolgen.

Definitionen und Nomenklaturen

Für die Ablage der Artefakte der Software-Entwicklung ist es hilfreich, sich im Vorfeld eine Struktur zu überlegen, die für den Einsatzzweck wiederverwendbar ist. Nur so können Synergieeffekte effizient genutzt werden.

Ein möglicher Vorschlag für eine Struktur könnte z.B. so aussehen:

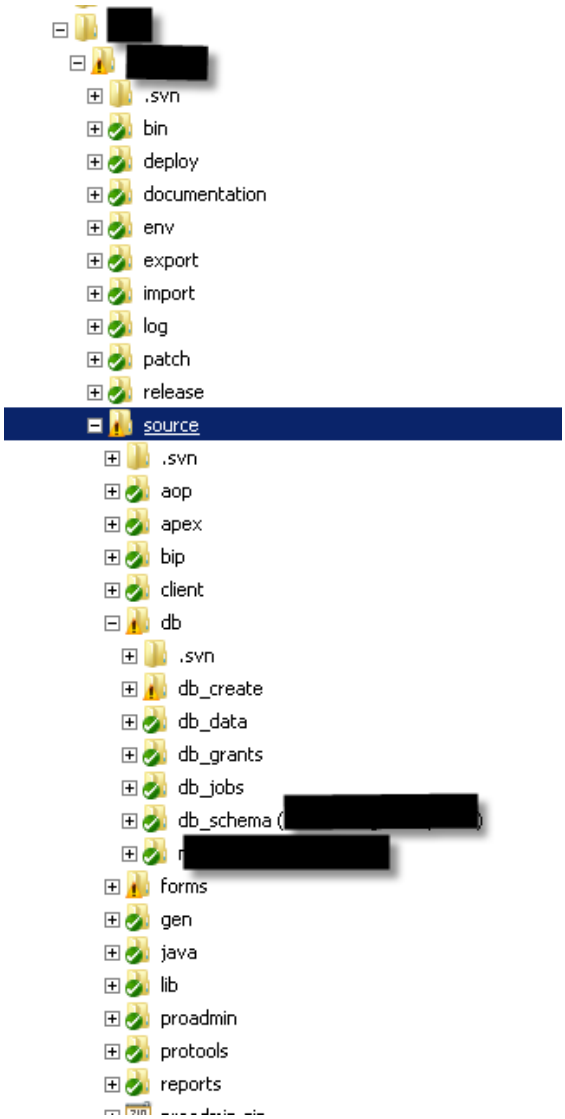

Abb. 1: Verzeichnisstruktur

Die Struktur ist über alle Projekte einheitlich und wird je nach verwendeter Software unterschiedlich genutzt. In der Struktur befinden sich bereits Artefakte, die einen allgemeingültigen Charakter haben. Skripte für das Deployment auf Server oder Datenbank, zum Starten oder Stoppen der Systeme, benutzte Bibliotheken, die Unternehmensweit zum Einsatz kommen.
Alle diese Sourcen sind über SVN-Externals in das Projekt eingebunden, so dass eine Erweiterung dieser allgemeingültigen Passagen allen Projekten zu Gute kommt.
Die Anpassung dieser Skripte an das jeweilige Projektumfeld (Servernamen, IP-Adressen, Benutzernamen, Passwörter, etc) werden an einer einzigen Stelle durch „Environment“-Files geführt.

Die Verzeichnisstruktur bereitet auch Skripte und Pfade für das Release- und Patchmanagement vor, so dass der Entwickler seine Sourcen nur an definierten Stellen ablegen muss, um den Patchvorgang zu automatisieren.

Datenmodell und Zugriffe

Innerhalb einer Firma entwickelt sich an allen Stellen eine Sammlung an Erfahrungen über die verschiedenen Projekte, die in so etwas wie ein „Best Practise“ zurückfließen. Optimal ist es, wenn dieses Wissen nicht nur in den Köpfen der Mitarbeiter steckt, sondern auch in nutzbare Artefakte zurückfließt und womöglich auch zwischen den Projekten weiter entwickelt wird. Beispiele für solche Artefakte können sein:

Low-level-Packages
An Hand des Datenmodelles erstellen wir skriptgesteuert Packages, die sich um die DML-Statements für eine Tabelle kümmern. Die Skripte sind in der Lage, die Metadaten der Tabelle (incl. UK, PK Constraints und SEQ) auszulesen und daraus die LowLevel-Packages zu generieren.

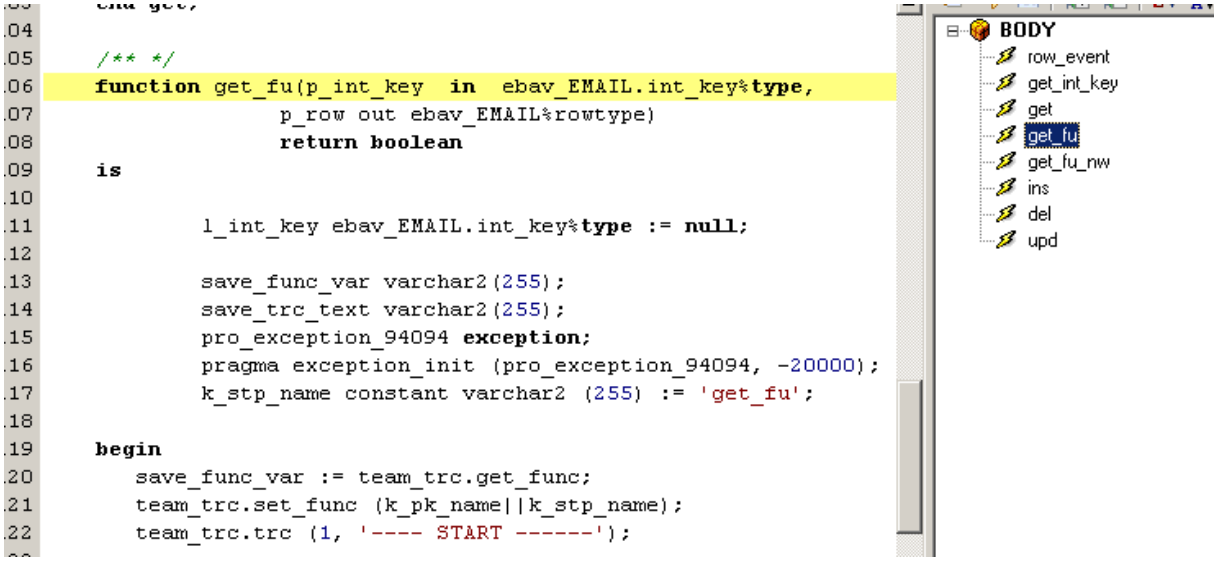

Abb. 2: Low-Level-Packages

Verfahren für Traces
Auch für die Ausgabe von Traces bietet es sich an, projektübergreifend einheitliche Standards zu definieren, die dann je nach Projekt unterschiedlich konfiguriert werden können. Dazu gehören die Unterscheidung, ob die Meldungen in eine Tabelle oder das File-System geschrieben werden und an welchen Stellen die Ausgaben ermöglicht werden sollen.

AOP
(Aspekt-orientierte Programmierung)
Nicht nur der standardisierte Einsatz der Trace-Meldungen, sondern auch z.B. die Sicherstellung der Cursor-Behandlung oder auch die Transaktionssteuerung können über diesen Ansatz umgesetzt werden.
Die Fragmente für diese Aufgabenstellung sind allgemeingültig erstellt und werden durch den PreCompiler in den PL/SQL-Code eingefügt. Dabei sind die Vorlagen pro Projekt konfigurierbar, so dass unterschiedlichen Aspekte umgesetzt werden können.

DDL-Framework
Ein weiteres Artekakt und Hilfsmittel gerade für den Einsatz eines optimierten Patchmanagement ist der Einsatz eines DDL-Frameworks. Dabei werden die Metadaten der Tabelle (Definition der Spalten) in einem eigenen Schema mitgeführt und bei DDL-Statements daraus der optimierte Code generiert. Ist eine Spalte noch nicht vorhanden, wird sie erstellt, ist die Spaltendefinition zu klein, wird sie erweitert.
Damit lässt sich auch für kleine Zeitfenster ein passendes Skript erstellen.

Applikationstemplates

Je nach eingesetzter Technologie (ADF, OWB, ODI, Forms, BI-Publisher) kommen weitere Artefakte zum Einsatz.
Als Beispiele dienen hier Historisierungsexperten für den OWB, Security-Bausteine für die ADF-Entwicklung, Bibliotheken für Forms.
Als diese Komponenten werden über SVN-Externals in die Projekte eingebunden und können so von Projekt zu Projekt weiterentwickelt werden.
Anpassungen erfolgen wo immer es möglich ist, durch Konfigurationsparametern an den Komponenten.

Deployment, Installation, Release und Patchmanagement

Für das Deployment sieht die Infrastruktur schon fertige Skripte für den WLS, den Glassfish und die Datenbank vor. Dabei können zum einen inkrementelle Updates oder auch eine komplette Installation ausgewählt werden.
Aufgabe des Konfigurationsmanagement an dieser Stelle ist die Sicherstellung der Abhängigkeiten zwischen den Versionen der eingesetzten Komponenten.
Produkte wie z.B. Maven können diesen Prozess begleiten, durchführen und auch dokumentieren.

Laufzeitkonfigurationen

Auch zur Laufzeit eines Produktes gibt es immer wieder Parameter, die das Verhalten der Anwendung pro Installation, Mandant oder sogar pro Benutzer steuern soll. Auf diesen zentralen Property-Store soll aus Sourcen der Datenbank, des Applikation Servers oder sogar des Client zugegriffen werden können.
Aus unserer Sicht empfiehlt sich der Aufbau einer hierarchischen Struktur ähnlich der Registry. Datenbank-Packages sichern den lesenden und schreibenden Zugriff auf diesen Store an Hand der Schlüsselpfade.

Fazit
Konfigurationsmanagement ist ein Invest in Wissen, Qualität und Performance. Achten sie auf Agilität und Verwendung. Jeder Ausbruch trägt einen Grund und im Zweifel einen guten, der den morgigen Tag effizienter gestalten lässt.

Kontaktadresse:
Stephan La Rocca
TEAM GmbH
Hermann-Löns-Str. 88
D-33104 Paderborn

Telefon: +49 (0) 5254-800865
Fax: +49 (0) 5254-800819
E-Mail sr@team-pb.de
Internet: www.team-pb.de