Projektmappen für effizientes Arbeiten richtig aufbauen

Ordnung ist das halbe Leben

Große erweiterbare Projekte, verschiedene Entwicklerteams, wiederverwendbare Komponenten, auswechselbare Technologien – eine Projektstruktur, die all dies bedenkt, spart Zeit und Aufwand.

Auf einen Blick

Boris Wehrle ist Senior Consultant bei der AIT GmbH & Co. KG. Er berät Unternehmen bei der Softwareentwicklung auf Basis von Microsoft-Technologien. Seine Schwerpunkte liegen in der Konzeption und Entwicklung verteilter Anwendungen im industriellen Bereich. Sie erreichen ihn unter Boris.Wehrle@aitgmbh.de.

Inhalt

- Die Dateistruktur als Basis für die Projektstruktur.
- Entwicklungszweige berücksichtigen.
- Einstellungen anpassen, um Aufwand zu reduzieren.

dnpCode
A1205Solutionstruktur

Anwendungen unterliegen im Laufe ihres Produktionslebens einer Reihe von sich verändernden Rahmenbedingungen. Neben fachlichen Anforderungen zählt hierzu unter anderem auch eine Weiterentwicklung der Technologie. An die Architektur einer Anwendung stellt dies eine Vielzahl von Anforderungen:

- Die Anwendung soll einfach und auf flexible Weise zu erweitern sein.
- Die verwendeten Technologien sollen sich austauschen lassen.
- Die Anwendung soll sich automatisiert testen lassen.
- Komponenten der Anwendung sollen wiederverwendbar sein.
- Die Anwendung soll von unterschiedlichen Entwicklern und Teams gleichzeitig zu bearbeiten sein.

Um dies zu erreichen, hat es sich unabhängig von den fachlichen Anforderungen bewährt, eine Anwendung nach dem Entwurfsmuster Model-View-ViewModel aufzubauen und dies mit einer dienstorientierten Architektur zu verbinden. Um den Entwicklungsprozess auch bei der dadurch bedingten höheren Komplexität produktiver gestalten zu können, ist darüber hinaus ein geeigneter Aufbau des Projekts notwendig, also seine Aufteilung auf Projekte und Projektmappen und deren Verwaltung. Dies hilft den Entwicklern, sich zurechtzufinden, und die häufig wiederkehrenden Prozesse wie das Kompilieren oder das Ausführen von Unit-Tests zu beschleunigen.

Damit all das nicht nur graue Theorie bleibt, soll ein Beispiel für einen Projektaufbau folgen, der sich bereits in einer Vielzahl von Projekten bewährt hat. Es handelt sich dabei um einen Ausschnitt aus einem Projekt des Systemhauses AIT, der unter [1] unter dem Namen NetFactory als Download zur Verfügung steht; auf der Heft-CD ist das Beispiel ebenfalls zu finden.

Die Voraussetzungen

Die Basis einer Anwendung ist eine klare Struktur im Dateisystem. Bei der Benennung der Verzeichnisse gilt es stets darauf zu achten, einen kurzen und zugleich aussagekräftigen Namen zu verwenden. Dies erleichtert die Orientierung und berücksichtigt, dass ein Dateipfad unter Windows maximal 260 Zeichen lang sein darf.

Um ein paralleles Arbeiten von mehreren Entwicklern oder Teams an der Anwendung zu ermöglichen, ist ein Versionskontrollsystem erforderlich, zum Beispiel der Team Foundation Server (TFS). Dieses ermöglicht nicht nur, unterschiedliche Versionsstände einer Anwendung zu verwalten, sondern auch verschiedene Entwicklungs- oder Versionszweige, sogenannte Branches. Der Visual Studio Branching Guide erklärt, wie sich der Umgang mit Entwicklungszweigen mit dem TFS gestaltet [2].

Das Arbeiten mit Entwicklungszweigen erlaubt es, die Anwendung weiterzuentwickeln, während zum Beispiel eine Produktivversion (Release) stabilisiert wird. Versionszweige sind einfacher zu verwenden, wenn alle zu einer Anwendung gehörenden Komponenten – Quellcode, Datenbanken, externe Bibliotheken et cetera – im Projektverzeichnis abgelegt werden. Hierzu wird auf dem TFS eine Verzeichnisstruktur nach folgendem Muster aufgebaut: Das Teamprojekt erhält ein Verzeichnis für das Projekt, in dem sich drei Unterverzeichnisse befinden – ein Verzeichnis für den Main-Entwicklungszweig, in dem sich später das eigentliche Projekt, also der Quellcode, befindet; ein Verzeichnis für den Feature-Zweig und eines für den Release-Branch. **Abbildung 1** zeigt die Struktur des NetFactory-Projekts, das nach diesem Schema aufgebaut ist. Alle Projektzweige befinden sich auf der gleichen Hierarchiestufe, dies vereinfacht die Navigation.

Die Basisstruktur für die Quellen

Im Projektverzeichnis – in **Abbildung 1** *NetFactory/NetFactory/Main/NetFactory* – wird die Anwendung in die eigentlichen Quellcodes, die Bibliotheken, Skripte et cetera verteilt. Dazu gibt es die entsprechenden Ordner wie *Libraries*, *Scripts*, *Sources* und so weiter.

Im Rahmen des Kompiliervorgangs entsteht auch ein *Bin*-Verzeichnis. Sein Inhalt spiegelt die auf dem Zielsystem zu installierenden Assemblies und Daten wider. Es bildet damit die Basis, auf deren Grundlage das Setup erzeugt wird. Zudem wird die Anwendung während des Debuggings in diesem Verzeichnis gestartet. Das Verhalten dabei entspricht dann in etwa