Vergleich von objektrelationalen Mappern

Florian Amstutz

Zürcher Hochschule für Angewandte Wissenschaften (17. Juni 2013)

Objektorientierte Programmiersprachen kapseln Daten und Verhalten in Objekten, relationale Datenbanken verwenden hingegen Tabellen. Um zwischen diesen grundverschiedenen Paradigmen übersetzen zu können werden objektrelationale Mapper (kurz OR-Mapper) verwendet.

Heutige serviceorientierte Architekturen verwenden fast in jedem Fall einen Persistenzlayer welcher Objekte in eine Datenbank speichert und sie von dort liest. Diese Aufgabe wird meist nicht von der Applikation selber übernommen sondern durch einen OR-Mapper ausgeführt. Es gibt eine Reihe von Einflussfaktoren und Entscheidungskriterien die beim Entwurf und der Implementierung von Persistenzkomponenten beachtet werden müssen. Die Auswahl des passenden OR-Mappers für die jeweilige Architektur ist eine der zentralen Fragen, die zu Beginn des Implementierungszyklus einer Applikation beantwortet werden muss (nach [Sta11]), denn oft ist der Persistenzlayer das Nadelöhr einer Enterprise Applikation (gemäss [LSL+07]).

Diese Bachelorarbeit erhebt und dokumentiert die Anforderungen an den Persistenzlayer einer serviceorientierten Architektur und ausgewählten Enterprise Pattern (nach [FRF+02]). Es wird eine Marktübersicht aller frei verfügbaren oder Open Source OR-Mapper im .NET-Umfeld erstellt und ausgewählte Produkte werden auf Grund einer Auswahl von Kriterien miteinander verglichen. Die gewählten OR-Mapper werden gegen die erhobenen Anforderungen geprüft und es wird eine Produktempfehlung für jedes Enterprise Pattern erstellt. Benutzt ein Persistenzlayer ein Domain Model, so wird die Verwendung von Entity Framework empfohlen. Für Table Module und Active Record wird der Einsatz von Dapper empfohlen. Wenn die Geschäftslogik eines Persistenzlayers als Transaction Script implementiert ist, wird entweder Entity Framework, NHibernate oder Dapper empfohlen. Diese Produktempfehlungen und die für die Empfehlung verantwortlichen Vergleichsdaten werden anhand eines Proof of Concepts gewonnen. Eine serviceorientierte CRM-Applikation wird exemplarisch implementiert, indem Anforderungen erhoben werden, ein Konzept erstellt wird und nach der Implementierung eine Verifikation der implementierten Anforderungen durchgeführt wird.

Als Fazit erkennt die Arbeit zwei Typen von OR-Mappern: Leichtgewichtige und komplette OR-Mapper. Diese beiden Typen von ORM unterscheiden sich im Funktionsumfang und werden in unterschiedlichen Softwaresystemen eingesetzt. In naher Zukunft werden in Enterpriseapplikationen verstärkt Domain Models verwendet werden, da Entwickler es heutzutage gewohnter sind mit Objekten zu arbeiten als mit tabellarischen Daten. Auch der Trend in Richtung objektrelationaler Datenbankmanagementsysteme wird die Verwendung von komplexen Domänenmodellen fördern und eventuell sogar OR-Mapper überflüssig machen. Trotzdem sieht der Autor dieser Bachelorarbeit besonders bei bedienerfreundlichen OR-Mappern wie dem Entity Framework grosses Potential in naher Zukunft. Auch leichtgewichtige OR-Mapper wie Dapper werden in performancekritischen Systemen und in Systemen mit einer einfacheren Domänenlogik eine Daseinsberechtigung haben und wohl noch an Bedeutung gewinnen. Wird sich Dapper in den Möglichkeiten des Monitorings und im Bedienungskomfort näher an die kompletten OR-Mapper wie NHibernate oder dem Entity Framework angleichen, ist der Autor durchaus zuversichtlich, dass in dieser Nische ein Potential für Dapper vorhanden ist.

Literatur

[FRF+02] Fowler, Martin ; Rice, David ; Foemmel, Matthew ; Hieatt, Edward ; Mee, Robert ; Stafford, Randy: *Patterns of Enterprise Application Architecture.* Addison Wesley, 2002

[LSL+07] Liebhart, Daniel ; Schmutz, Guido ; Lattmann, Marcel ; Heinisch, Markus ; Könings, Michael ; Kölliker, Mischa ; Pakull, Perry ; Welkenbach, Peter: *Architecture Blueprints.* Carl Hanser Verlag, 2007

[Sta11] Starke, Gernot: *Effektive Software-Architekturen.* Carl Hanser Verlag, 2011