1 UFO (Ultimate - Festival - Organizer)

Folgende Dokumentation stellt die Gesamtdokumentation der aufbauenden Übung UFO dar, die im Zuge der Realisierung iterativ über die drei Ausbaustufen hinweg erweitert wird.

1.1 Ausbaustufe 1 (ADO.NET)

Folgender Teil dokumentiert die erste Ausbaustufe der aufbauenden Übung UFO. In diesem Teil wird die Persistenz Schicht in .NET unter Hilfenahme von ADO.NET implementiert. Aufgrund der Analyse der Gesamtaufgabenstellung wurde entschieden das vorerst nur die Persistenz Schicht an sich, also INSERT, UPDATE, DELETE, der einzelnen Entitäten realisiert wird, da die Geschäftslogik erst bei der Realisierung der Administration und des Webzugriffs endgültig feststehen wird.

Im Zuge der Realisierung des Webzugriffs wird auch der Web-Service implementiert werden müssen, der die Daten der Web Applikation zur Verfügung stellt. Dieser soll die Daten bereits gefiltert und strukturiert zur Verfügung stellen, daher besteht eine gewisse Abhängigkeit zwischen dem Web-Service und der Web Applikation sowie auch der Client Administration.

Daher wird die Web-Service Implementierung die eigentliche Geschäftslogik enthalten, die in einer Transaktion abgearbeitet und im wesentlichen aus den logischen Prüfungen gegen die Datenbank bestehen wird. Die einzelnen Datenabfragen, die benötigt werden können einfach hinzugefügt werden. Diese Geschäftslogik soll über eine REST-API zur Verfügung gestellt werden, die spezifiziert, wie mit dem System zu interagieren ist, daher stellt dieser Web-Service die Kernkomponente des Gesamtsystems dar.

1.1.1 Systemaufbau

Folgend ist der Systemaufbau der Persistenz Schicht dokumentiert.

Die folgende Auflistung illustriert dir Projektstruktur der Persistenz Schicht:

1. *UFO.Server.Data.Api*
   Dieses Projekt enthält die Spezifikation der Persistenz Schicht in Form von Interfaces, abstrakten Klassen, Exceptions und den Entitäten.

2. *UFO.Server.Data.MySql*
   Dieses Projekt enthält die MySql spezifische Implementierung der Persistenz Schicht.

3. *UFO.Server.Test.Data.MySql*
   Dieses Projekt enthält die MySql spezifischen Tests der Persistenz Schicht.

4. *UFO.Common.Util*
   Dieses Projekt enthält die Utlities für die UFO Infrastruktur in .NET, die nicht spezifisch einen Systemteil zuzuordnen ist.

Alle .NET Systemteile werden unter dem Namensraum *UFO.** zusammengefasst.

Die folgende Auflistung illustriert die verwendeten Technologien und Frameworks:

1. *MySql*
   Es wird eine MySql Datenbank verwendet, die Open-Source ist und eine Integration in .NET besitzt.

2. *NUNIT*
   Als Test-Framework wird NUNIT verwendet, da es mehr Funktionalität mitbringt als das Standard Test-Framework integriert in .NET.

3. *ADO.NET*
   Als Persistenz Provider wird wie gewünnscht ADO.NET und kein ORM Mapper verwendet.

1.1.2 Datenbank

Es wurde als Datenbank MySql und Moderlierungstool MySql Workbench gewählt, da diese Datenbank erstens Open-Source, zweitens eine gute .NET Integration vorhanden ist sowie drittens bereits Erfahrungen mit dieser Datenbank vorhanden waren.

Das folgende ER-Diagramm illustriert das implementierte Datenbank Schema.

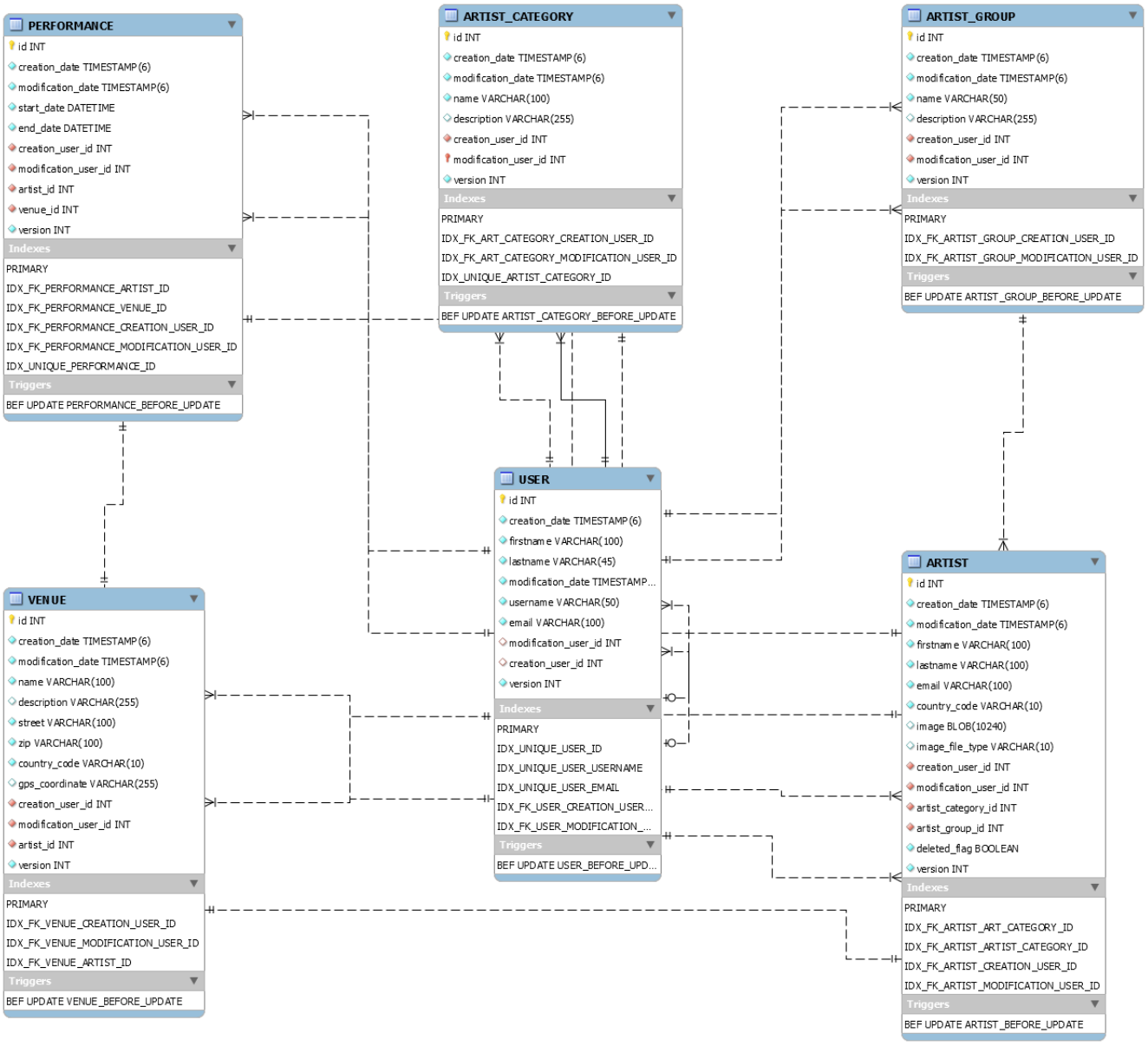

Abbildung 1: ER-Diagramm des Schema 'UFO'

Auf jeder Entität wurde eine Spalte für die Versionierung eingeführt (MySqlDbType.LONG), die über einen Update-Trigger bei jedem Update um eins erhöht wird sowie auch das Modifizierungsdatum. Ein ganzahliger Datentyp erschient hier mehr sinnvoll, da es hier mit Sicherheit keine Kollisionen geben kann, nicht so wie bei einem Timestamp Datentyp.

Ebenso halten alle Entitäten eine Referenz auf den Benutzer der Sie erstellt sowie zuletzt modifiziert hat. Dies dient der Nachverfolgbarkeit von Änderungen, zumindest wer zuletzt eine Änderung durchgeführt hat.

1.1.3 Klassenhierarchien

Folgend ist die Struktur der Klassenhierarchien dokumentiert.

IDao

Folgendes Klassendiagramm zeigt die Hierarchie des Interfaces *IDao*, das der Basistyp für alle implementierten DAO Interfaces dient, da es bereits alle Basisaktionen auf eine Entität definiert.

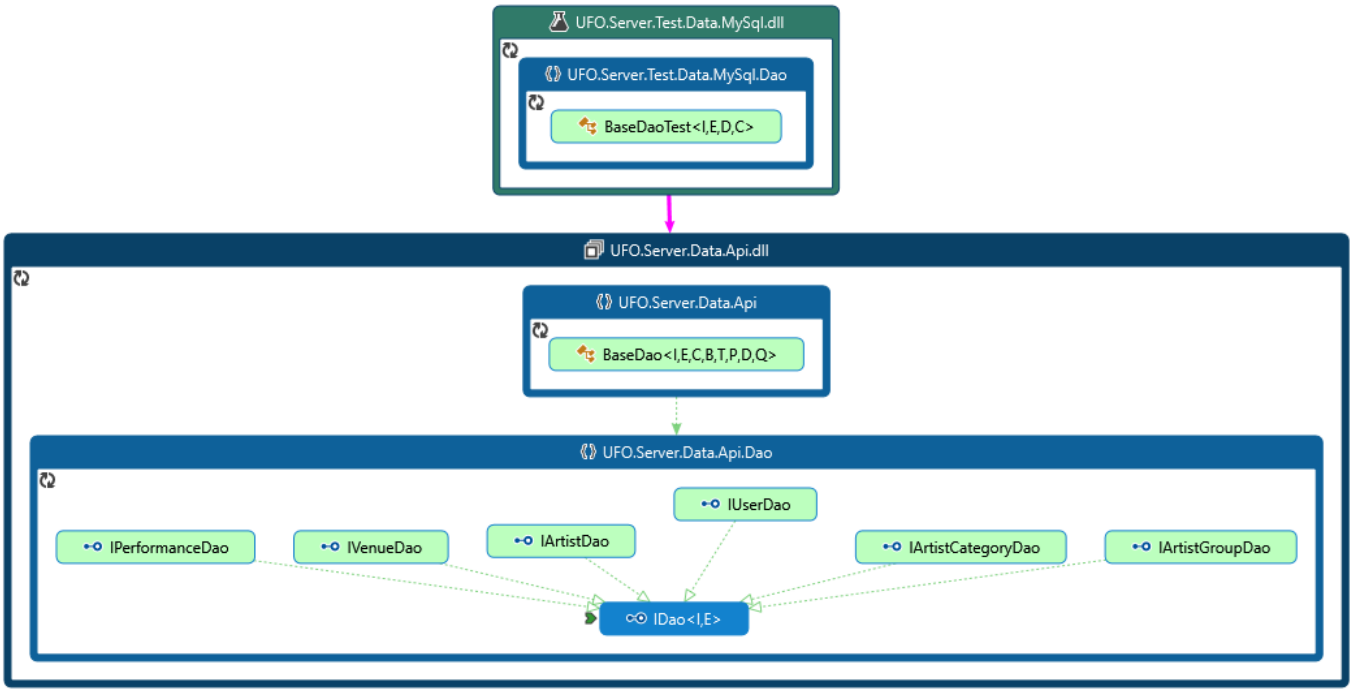

Abbildung 2: Klassenhierarchie IDao Interface

Die Basisklasse *BaseDao* implementiert alle Methoden, die in *IDao* definiert wurden für alle implementieren Entitäten sofern sie *IEntity* implementieren. Dieser generische und abstrakte Ansatz erlaubt es dass die Basisfunktionalität eines DAOs nur einmal für alle Entitätstypen, die *IEntity* implementieren, implementiert werden musste.

IEntity

Folgendes Klassendiagramm illustriert die Klassenhierarchie des Interfaces *IEntity*, welches den Basistyp für alle Entitäten darstellt.

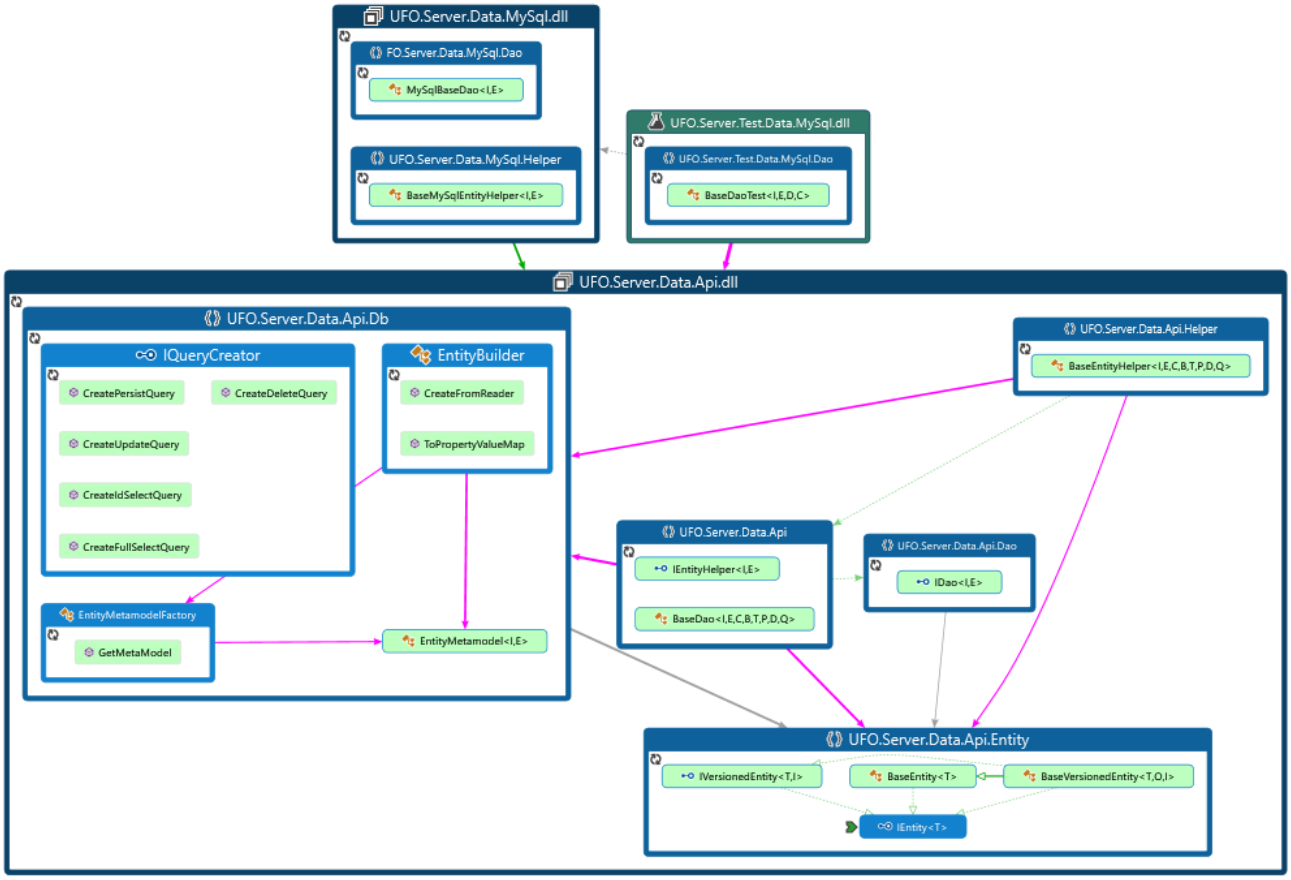

Abbildung 3: Klassenhierarchie IEntity Interface

Es wurden Basisentitäten eingeführt, welche eine Basisstruktur definieren dies sich Entitäten unterwerfen müssen wenn sie von diesen Klassen ableiten. Somit wird eine konsistente Struktur der Entitäten bzw. deren Tabellenrepräsentation gewährleistet.

IEntityHelper

Folgendes Klassendiagramm illustriert die Klassenhierarchie des Interfaces *IEntityHelper*. *IEntity*, welches den Basistyp für alle Entitäten darstellt.

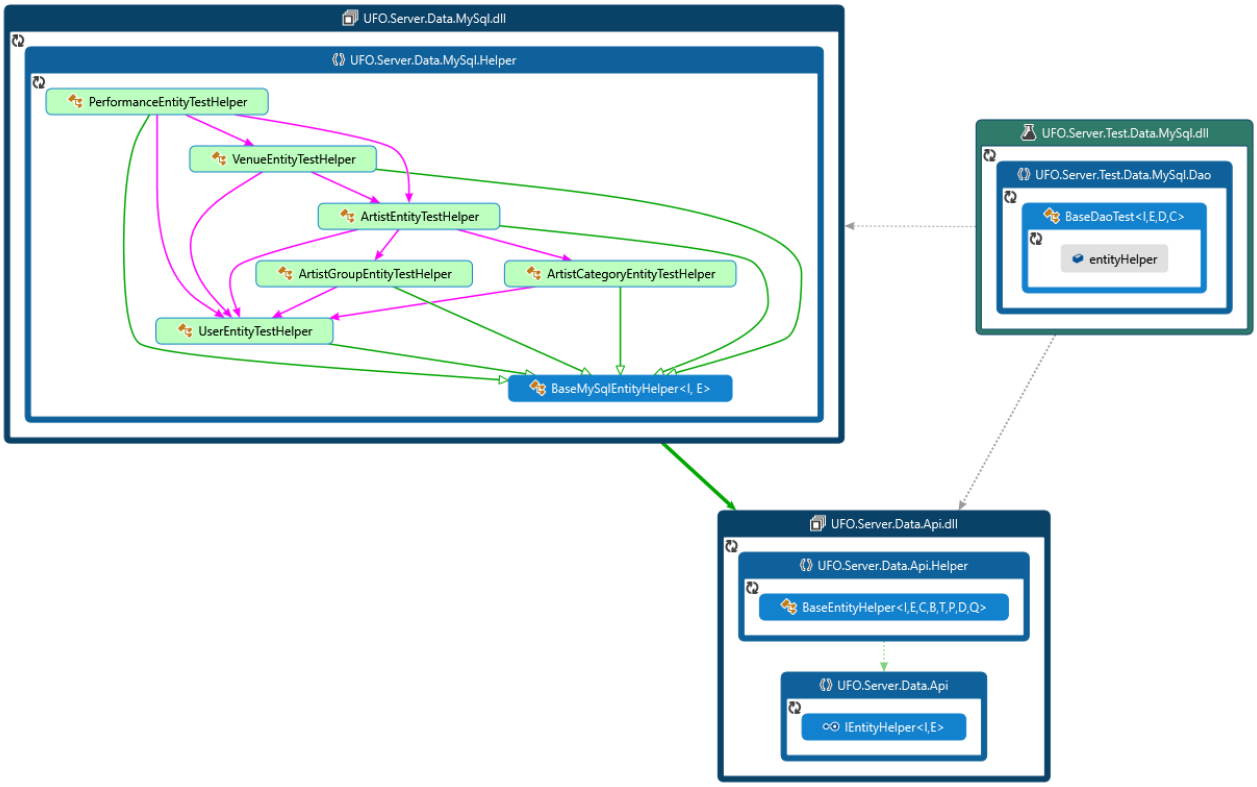

Abbildung 4: Klassenhierarchie IEntityHelper Interface

Dieses Interface und dessen Implementierungen diesen als Hilfestellung für Test und die Generierung von Testdaten über die Entitäten Modelle. Würde hier die unabhängige Persistenz Logik in *BaseEntityHelper* auh noch ausgelagert werden do könnte man diese Hiflsklassen ebenfalls in das Projekt *UFO.Server.Data.Api* überführt werden, da hier nur in der Persistenz Logik die Abhängigkeiten zum Persistenz Provider enthalten sind. Diese Hilfklassen entstanden aufgrund der generischen Testklasse *BaseDaoTest*, die die Entitäten nicht erzeugen kann und daher diese außerhalb geschehen muss.