Unterstützung von Object Relational Mapping

Szenarien im Projekt

- Verwendung eines existierenden Datenmodells
- Datenmodell wird im Projekt (zusammen mit dem fachlichen Modell) entworfen

Art des Datenmodells bzgl. Delta-Historie

- Delta-Historie
- keine Delta-Historie

Konsequenz aus Verwendung Delta-Historie

Die Fachklassen können nicht direkt als Entities (im JPA-Sinne) verwendet werden.
Wieso? Weil eine Entity (Objekt) in Abhängigkeit davon, ob es in der aktuellen Historie geändert wurde oder nicht in unterschiedlichen Zeilen in der DB gespeichert wird. Dies lässt sich mit den Konzepten in Hibernate & JPA nicht abbilden.

Statt Delta-Historie ja/nein, könnte man allgemeiner unterscheiden

- Fachmodell lässt sich mit den bestehenden JPA/Hibernate Konzepten direkt auf die DB Tabellen mappen.
- Fachmodell lässt sich nicht direkt mit den bestehenden JPA/Hibernate Konzepten auf die DB Tabellen mappen.

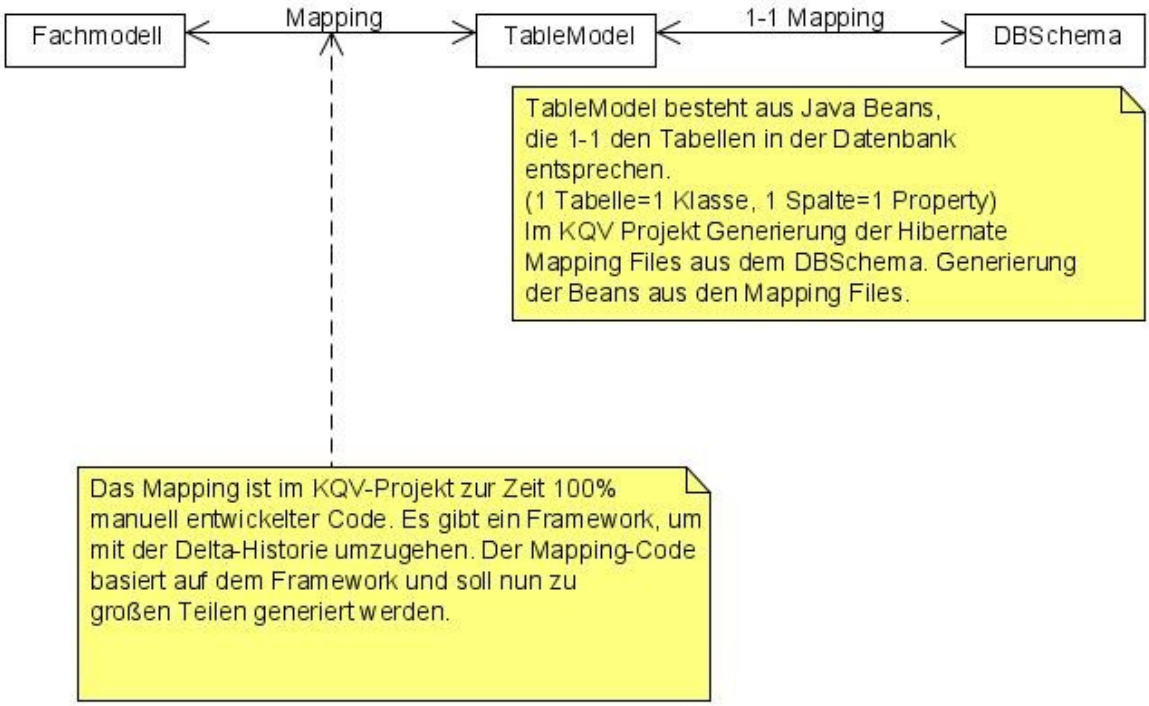
Art des Datenmodells bzgl. Unterstützung „Dynamik Properties“

- Jedes Property im Fachmodell wird in einer eigenen Spalte in der DB gespeichert.
- Unterstützung von Dynamik Properties in der Datenbank. Z. B. durch eine Tabelle PROPERTYVALUE mit den Spalten PROPERTY_NAME und PROPERTY_VALUE.

Anforderung KQV-Projekt

- Austauschbarkeit der Persistenz
  Grund hierfür: Verwendung des gleichen fachlichen Modells im Angebot & Vertragssystem aber mit unterschiedlicher Persistenz.
- Persistenz wird in eigenen Eclipse-Projekten definiert. Die Persistenzprojekte sind abhängig von den Fachmodell-Projekten (nicht umgekehrt).
- Existierendes Datenmodell mit Delta-Historie
- Unterstützung Datenmodell mit „INPU-Logik“ (=die KQV spezifische Ausprägung von Dynamic Properties in der DB).

Überblick KQV Lösung: Existierendes Datenmodell mit Delta-Historie und Dynamic Properties Support.



Zusammenfassung Scope

Faktor-IPS soll das Mapping der Fachmodelle auf relationale Datenbanken unterstützen.
Das Datenmodell kann entweder bereits existieren oder zusammen mit dem Fachmodell erstellt werden. Arbeitet man mit einem existierenden Datenmodell sollte das Mapping gegen dieses Datenmodell geprüft werden (Gibt es die im Mapping angegebenen Tabellen/Spalten?).
(Hierzu müssen wir in Faktor-IPS ein Metamodell für Tabellen hinterlegen, und einen Mechanismus, mit dem die Informationen aus anderen Quellen eingelesen werden können).

Das Fachmodell kann mit den in JPA/Hibernate zur Verfügung gestellten Konzepten direkt auf das Datenmodell abgebildet werden oder (zum Beispiel bei Verwendung einer Delta-Historie) über den Umweg TableModel abgebildet werden. Es muss eine Unterstützung von Dynamic Properties geben, die (evtl. über Extensions) angepasst werden kann, sodass die bei der KQV unter der Bezeichnung INPU bekannte Variante unterstützt wird.
Für ein Fachmodell sollen verschiedene Persistenzmappings angegeben werden können, so dass das Fachmodell in unterschiedliche DB-Schemas persistiert werden kann.
Letztendlich bedeutet dies, dass wir das Mapping nicht zusammen mit den Fachklassen (=im selben File) abspeichern können, sondern das Mapping separat speichern müssen. Das sollte aber nicht dazu führen, dass der Benutzungskomfort beim Standardfall (es gibt genau ein OR-Mapping für ein Fachmodell) allzusehr darunter leidet.

In Faktor-IPS soll das OR-**Mapping** definiert werden können, nicht das komplette physikalische DB-Modell. Informationen zu Indizes, Partitionen, Tablespaces, etc. sind außerhalb des Scopes. Statt dessen kann man in einem weiteren Schritt über Schnittstellen zu entsprechenden Tools nachdenken.