Мухина А.Е.

Аспекты информационной безопасности при использовании технологии Object Relational Mapping

Аннотация: проанализированы основные подходы при использовании технологии Object Relational Mapping. Выявлены основные плюсы и минусы каждого подхода. Дана оценка подходам с учетом требований к информационной безопасности при реализации программного обеспечения.

Ключевые слова: информационная безопасность, объектно-ориентированное программирование, Entity Framework

Технологии в IT-индустрии продолжают развиваться, не сбавляя оборотов. Во многом, большинство новых открытий отрасли направлены на улучшение жизнедеятельности людей, благодаря автоматизации тех или иных сфер. Но не всегда разработчики, которые создают новый продукт, ориентируются на потребителей вне IT отрасли – многие специалисты разрабатывают такие продукты, которые помогают облегчать процесс создания программ.

Одним из основных фреймворков, который сейчас применяется при написании программ с использованием объектно-ориентированных языков, является Object Relational Mapping (ORM). Данная библиотека позволяет бороться с «потерей соответствия» – ситуацией, когда данные могут быть искажены, поскольку процедурные языки и языки запросов баз данных основаны на разной семантике и стратегии оптимизации, что приводит к несоответствиям между виртуальными объектами и прототипами [1]. Разработчику, который использует такой подход в своей работе, больше не требуется писать SQL-код для взаимодействия с СУБД, так как ORM-решение автоматически преобразовывает структуру классов в данные, которые могут храниться в системе управления базой данных.

ORM позволил в значительной степени увеличить скорость разработки приложений, но учитывает ли этот подход проблемы достоверности информации при проектировании и эксплуатации баз данных? Для примера, рассмотрим одну из наиболее

популярных библиотек, который позволяет работать с языком C# - Entity Framework. При работе с данной библиотекой, зачастую приходится выбирать один из трех доступных подходов к разработке приложений: Model-first, Database-first, Code-first [2].

Первый подход (Model-first) позволяет сформировать базу данных непосредственно после моделирования её в графическом редакторе. Созданная схема будет использоваться ORM для автоматической генерации скрипта SQL и построения базы данных, а также для формирования файлов исходного кода модели данных. Model-first позволяет сократить время разработки БД, особенно когда структура данных значительна. При очевидности плюсов, автоматически сгенерированный SQL-код может приводить к потерям данных в случае обновления структуры. Ведь при попытке внесения изменений вручную, при каждом изменении модели, все будет пересоздаваться заново.

Предполагается, что очевидную проблему позволит исправить второй подход (Database-first), при котором разработчик сначала создает SQL скрипты для построения базы данных, а ORM генерирует модель данных и классы приложения. В данном случае, риск потери данных будет сведен к минимуму, но ручное обновление базы данных может быть сложным, если мы имеем дело с кластерами или несколькими средами разработки/тестирования, поскольку нам придется вручную синхронизировать их, а не полагаться на управляемые кодом обновления/миграции или автосгенерированные SQL скрипты.

Флагманским подходом является Code-first, при котором разработчик создает классы сущностей модели данных, а Entity Framework генерирует базу данных. При этом слой графической модели данных отсутствует в приложении, поэтому данный подход также не является эталонным при разработке ПО.

Все перечисленные выше подходы не позволяют в полной мере избежать снижения достоверности информации в БД на различных этапах жизненного цикла приложения. Так, при применении Code-first, пользователю сложно учесть все связи между элементами, ограничения предметной области. Особенно остро эта проблема проявляется при наличии обновлений, так как проект может быть большим, а никакого графического контроля и моделирования не предусмотрено. С другой стороны, при использовании такого

подхода, все манипуляции с базой данных происходят автоматически, что предупреждает несанкционированные модификации в БД. С применением Database-first, пользователь также может столкнуться с многочисленными ошибками неучтенных данных, не защищен от манипулирования данными в базе, но при этом, данные практически никогда не будут потеряны при модификациях, так как они будут происходить непосредственно через базу данных.

Новые подходы к разработке программного обеспечения, безусловно, помогают решать множество задач, связанных со скоростью написания приложения, с возможностями автоматического генерирования кода, но при этом, они зачастую не могут обеспечить синтез оптимальных структур базы данных или классов приложения с должным вниманием к достоверности и защите данных. При этом описанные в статье фреймворки продолжают постоянно развиваться, и с течением времени, описанные проблемы также смогут быть невелированы как, например, с помощью ORM-подхода, практически решены проблемы SQL-инъекций, которые совсем недавно были ключевой проблемой в обеспечении безопасности баз данных.

Литература:

1. *Левчук С. В.* Об интеграции реляционной и объектной моделей при разработке информационных систем / С.В. Левчук // Сборник научных трудов Новосибирского Государственного Технического Университета. – 2010. – № 2. – С. 89-98.
2. Entity Framework tutorial [Электронный ресурс]. – Режим доступа: https://entityframework.net/ef-code-first. - (Дата обращения: 27.10.2019)
3. Документация по Entity Framework [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/ru-ru/ef/. – (Дата обращения: 03.11.2019).