УДК 004.4

РАЗРАБОТКА АРХИТЕКТУРЫ ORM НА ПРИМЕРЕ ENTITYOBJECTORM

Соснин А.С.

Научный руководитель – к.т.н. Маглинец Ю.А.

Сибирский федеральный университет

При разработке широкого круга объектно-ориентированных приложений, будь то корпоративные информационные системы, хранилища медиа данных и пр., возникает задача организации хранения объектов в базе данных. В случае использования реляционной базы данных данная задача решается с использованием технологии объектно-реляционного преобразования (ORM). Существует множество реализаций данной технологии для различных языков программирования и баз данных. Рассмотрим их основные недостатки:

Недостаточная гибкость некоторых решений не позволяет отойти от стандартного алгоритма работы ORM, что иногда является камнем преткновения в развитии системы.

Отсутствие синхронизации данных на уровне выше уровня базы данных (БД). Если одни и те же данные меняются в разных приложениях одновременно, то в итоге в БД сохраняются только последние изменения.

Узкая специализация - некоторые ORM предназначены для использования только с какой-то одной БД.

К сожалению, не было найдено решения, в котором нет ни одного из перечисленных недостатков, поэтому было принято решения разработать новую систему. На основе обзора существующих решений были выработаны следующие требования:

18) Поддержка различных баз данных;
19) Возможность гибкой работы с БД;
20) Автоматическая генерация схемы базы данных;
21) Загрузка данных по требованию;
22) Единовременное сохранение всех изменений в данных;
23) Решение коллизий одновременного редактирования данных.

Параметры отображения свойств объекта к полям базы данных задается в конфигурации. Конфигурация также предназначена для хранения информации об отношение между объектами (1-n, n-n). Конфигурация может храниться непосредственно в базе данных или XML файле. Первый вариант хорош для уже готовой системы. Второй же лучше подходит для процесса разработки.

На основе конфигурации автоматически генерируются запросы к базе данных на загрузку и сохранение данных. В случае если технология ORM применяется уже на существующую БД это не всегда возможно. В этом случае необходимо предусмотреть возможность выбора между автоматически построенными запросами и написанными вручную. Для того чтобы ORM могла корректно использовать вручную написанные запросы они должны иметь специальные метки для подстановки значений.

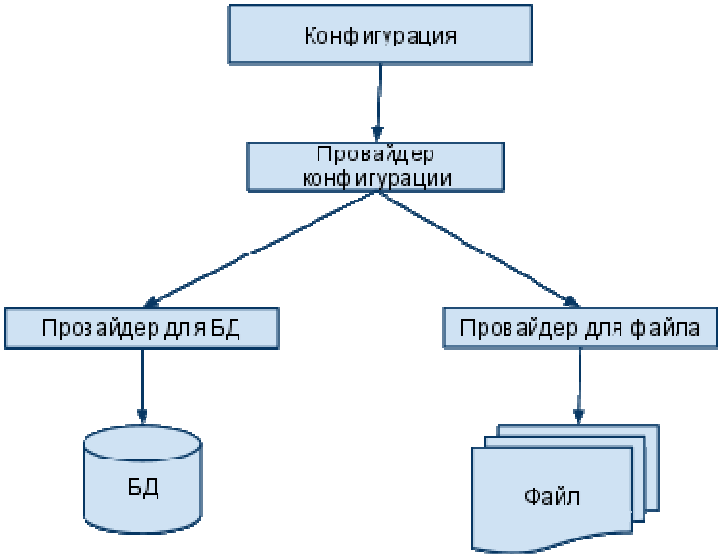

Рисунок 1 - Структура взаимодействия системы с конфигурацией

При использовании системы удобнее опираться на типы данных языка программирования, а не на типы данных конкретной БД. У каждой базы данных есть свои особенности и ограничения по встроенным типам данных. Для того чтобы абстрагироваться от конкретной БД для каждой из них необходим свой провайдер, который уже заботится о том, каким образом отображать типы данных языка программирования в типы данных БД и наоборот.

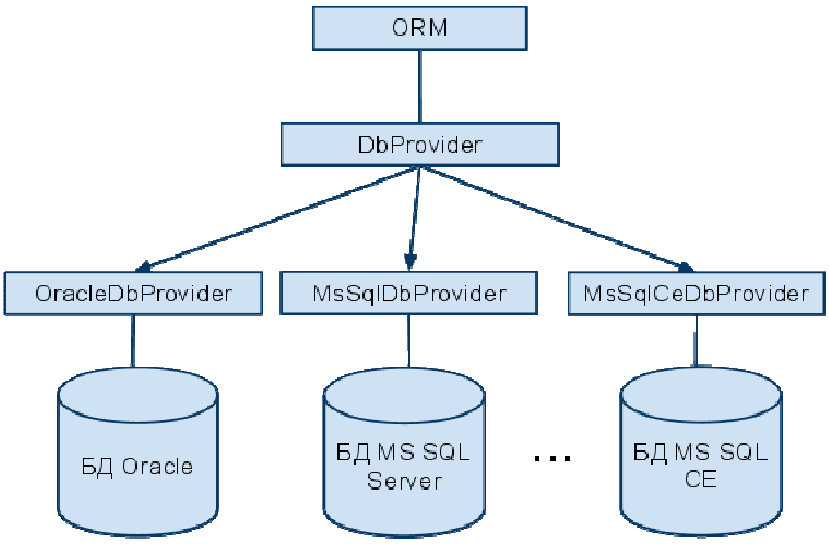

Рисунок 2 - Структура взаимодействия системы с различными БД

Так же дополнительной особенностью разрабатываемой системы является то, что она позволяет сократить рутинные операций при создания новой БД и последующего применения на ней технологии ORM т.к. схема БД может генерироваться автоматически в процессе настойки конфигурации.

Разрабатываемая система поддерживает механизм сессий. Изменение данных в одной сессии никаким образом не влияют на данные другой до сохранения первой.

Данные в процессе работы системы кешируются, что позволяет загружать их в разных сессиях не обращаясь повторно к БД.

В процессе работы системы в многопользовательской среде возможно возникновение такой ситуации, что какие-то данные одновременно пытаются редактировать два пользователя. В этом случае необходимо предоставить средство детектирования таких ситуаций. При возникновении такой ситуации необходимо разрешить редактирование минимальной порции данных только одному пользователю. Второй пользователь получит возможность записи данных только после отмены или сохранения изменений первым пользователем. Если данные были изменены и сохранены, то системе необходимо перезагрузить их снова для получения актуальной версии. Все эти функции выполняет специальный сервер блокировок доступ к которому клиентские приложения получают по сети.

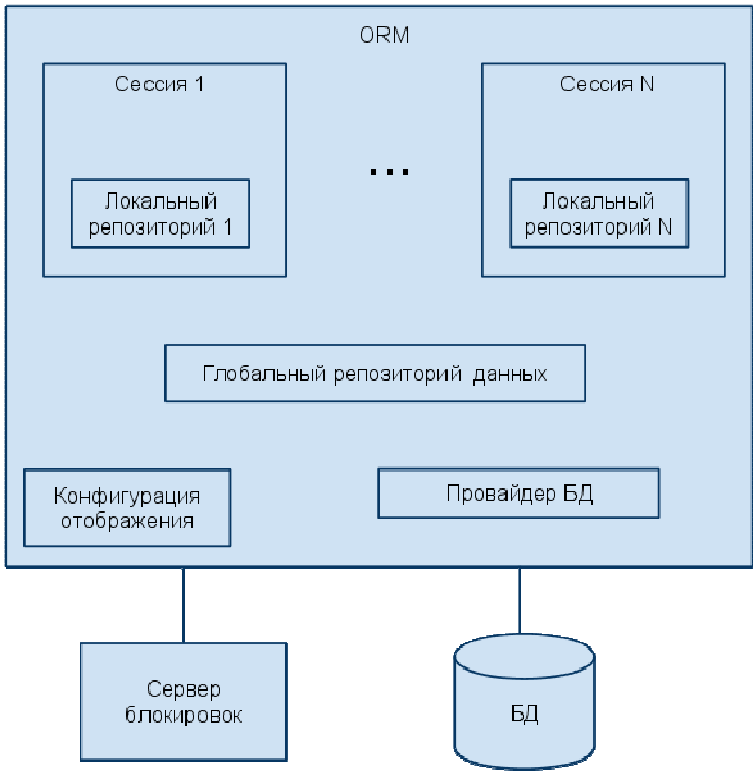

Рисунок 3 - Обобщенная архитектура системы

На основе описанной структуры была реализована первая версия данной ORM, которая успешно апробирована в проекте “Автоматизированная система приема и хранения космических снимков (АСПКС)” в институте космических и информационных технологий Сибирского федерального университета. Проект ORM получил название EntityObjectORM.